NRL FR-7841

Tracking System for Two Asynchronously Scanning Radars

Ben H. Cantrell
Gerald V. Trunk
Jon D. Wilson

December 5, 1974

Naval Research Laboratory
Washington D.C. 20362

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: NRL Report 7841	2. GOVT ACCESSION NO. / 3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle): TRACKING SYSTEM FOR TWO ASYNCHRONOUSLY SCANNING RADARS	5. TYPE OF REPORT & PERIOD COVERED: Interim report on a continuing NRL problem
	6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s): Ben H. Cantrell, Gerard V. Trunk, and Jon D. Wilson	8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Naval Research Laboratory, Washington, D.C. 20375	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: NRL Problem R02-54.101, 62712N, RF 12-151-403-4010
11. CONTROLLING OFFICE NAME AND ADDRESS: Department of the Navy, Office of Naval Research, Washington, D.C. 20375	12. REPORT DATE: December 5, 1974
	13. NUMBER OF PAGES: 87
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)	15. SECURITY CLASS. (of this report): Unclassified
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

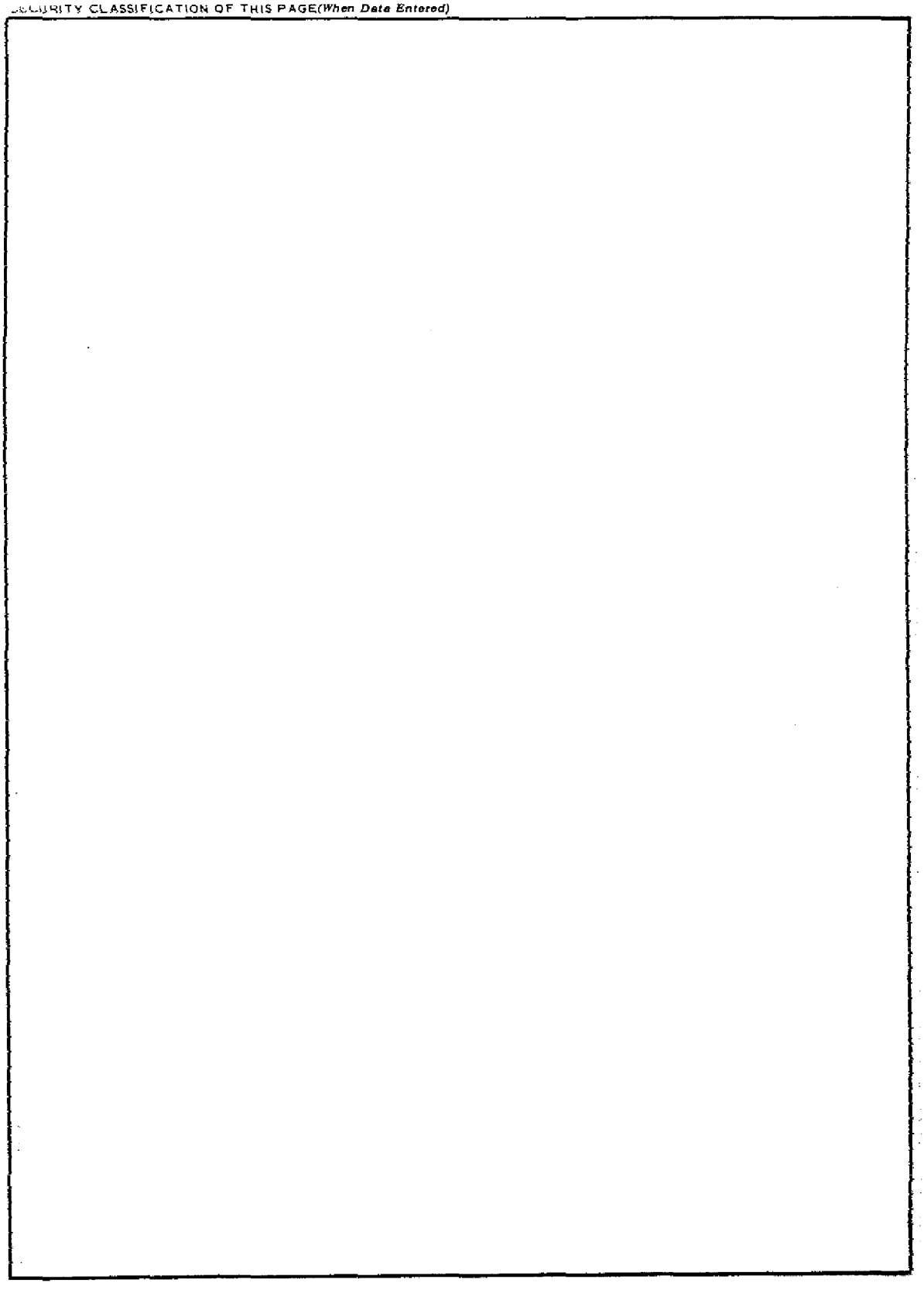
18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Computer
Search radar track
Tracking
Tracking algorithms

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

A tracking system has been designed to use detections from two asynchronously scanning radars located in close proximity. The system differs from previous single-radar tracking systems in timing, filter updating, and track initiation, and in using detections from two radars. Simulation results show that the system operating on 100 clutter points and 50 targets requires about 3.0 s using a Fortran program on a Nova 800 computer.



CONTENTS

TRACKING SYSTEM FOR TWO ASYNCHRONOUSLY SCANNING RADARS

1.0. INTRODUCTION

In the past a number of automatic detection and tracking systems, each using detections from a single radar, have been constructed. This report describes a tracking system that uses detections from two asynchronously scanning radars located in close proximity. The radars used are the SPS-12 and SPS-39; provisions have been made to add the SPS-10 at a later date. The general configuration is shown in Fig. 1. The detection and measurement procedures for each radar are described in Refs. 1 and 2. In addition, Ref. 1 describes the general operation of the system. The SPS-12 is a two-dimensional (2-D) radar with a scan period of 6 s. The SPS-39 is a three-dimensional (3-D) radar with a scan period of 8 s which operates in a special mode using only two beams and thus acts as a 2-D radar. This is used to decrease the multipath fading when both radars are considered together. Also, the operator can ask for height information on a specified target and then the SPS-39 will perform an elevation scan over a small sector in azimuth about the target [1].

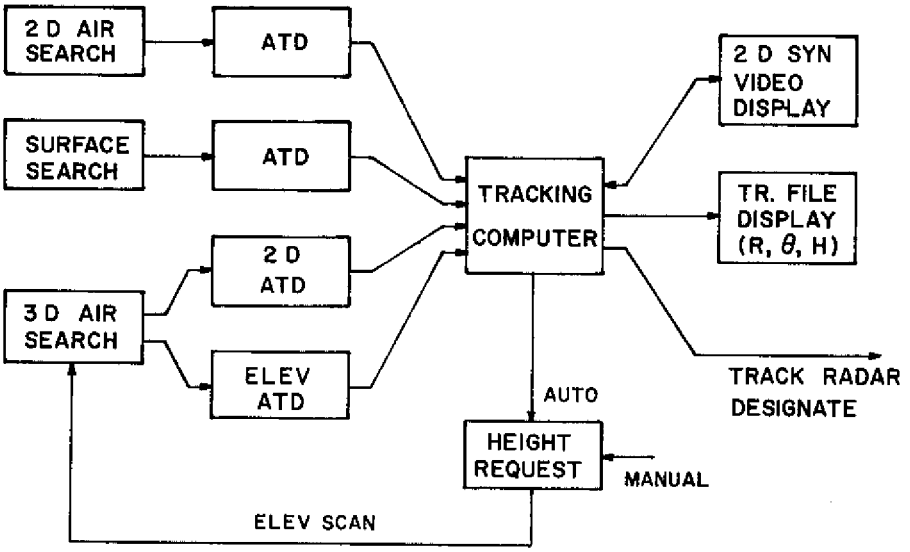


Fig. 1 — General configuration of the system

The tracking system, which is the topic of this report, resides in a minicomputer (a Data General Nova 800).

Three types of tracks are considered: clutter points (or slowly moving targets), target tracks, and tentative (or new) tracks. The tracks are correlated or associated with the detections from the radars. The tracks are smoothed, and each target's position is predicted for the next time the radar will be over the target. In order to reduce the number

Note: Manuscript submitted October 29, 1974.

of correlations to be performed, the tracks are stored in 64 sectors, and only those detections in the sector where the track is located and in neighboring sectors need be considered.

Most of the single-radar tracking systems use the radar itself for a clock, since the radar operates at a constant scanning rate. While this system is similar to other tracking systems using a single radar, it differs from previous single-radar tracking systems in timing, filter updating, and track initiation, and in the use of detections from two radars. Section 2 defines the basic system parameters and discusses the basic routines.

2.0. TRACKING SYSTEMS STORAGE FILES AND BASIC ROUTINES

When a track is established in the software of the computer, it is convenient to assign it a track number. With this system, all parameters associated with a given track are referred to by this track number. Each track number is also assigned a sector (region of space in azimuth) such that the correlation process (described in Secs. 4 and 5) can be performed efficiently. In addition to the track files, a clutter map is maintained. A clutter number is assigned to each stationary or very slowly moving target. All parameters associated with a clutter point are referred to by this clutter number. Again, each clutter number is assigned a sector in azimuth for efficient correlation.

The input data bank, which includes detection measurements and control parameters from the radars, is described in section 2.4. Most of the remaining parameters are listed, along with the previously defined parameters, in section 2.6. Finally, two short routines are described: the smoothing filter and the calculation of detection time.

2.1. Track and Clutter Number Files

The track and clutter number files are the same as those described by Richeson of APL [3]. The parameters required for the files are listed below.

Parameters	Description
NT	Track number
DROPT	1 (obtain) or 0 (drop) a track number NT
FULLT	Number of available track numbers
NEXTT	Next track number available
LASTT	Last track number not being used
LISTT (256)	File whose 256 locations correspond to track numbers
NC	Clutter number
DROPC	1 (obtain) or 0 (drop) a clutter number NC
FULLC	Number of available clutter numbers
NEXTC	Next clutter number available
LASTC	Last clutter number not being used
LISTC (256)	File whose 256 locations correspond to clutter numbers

Only the operation of the track number file is described, since the clutter number file's operation is identical.

The track number file is begun by setting LISTT (I) = I + 1 for I = 1 through 255. LISTT (256) is set equal to zero (denoting the last available track number in the file), NEXTT = 1 (the next available track number), LASTT = 256 (the last track number not being used), and FULLT = 255 (indication that 255 track numbers are available). A flowchart of the operation is shown in Fig. 2, and the subroutines TRKNO (NT, DROPT) and CLTNO (NC, DROPC) appear in Appendix A.

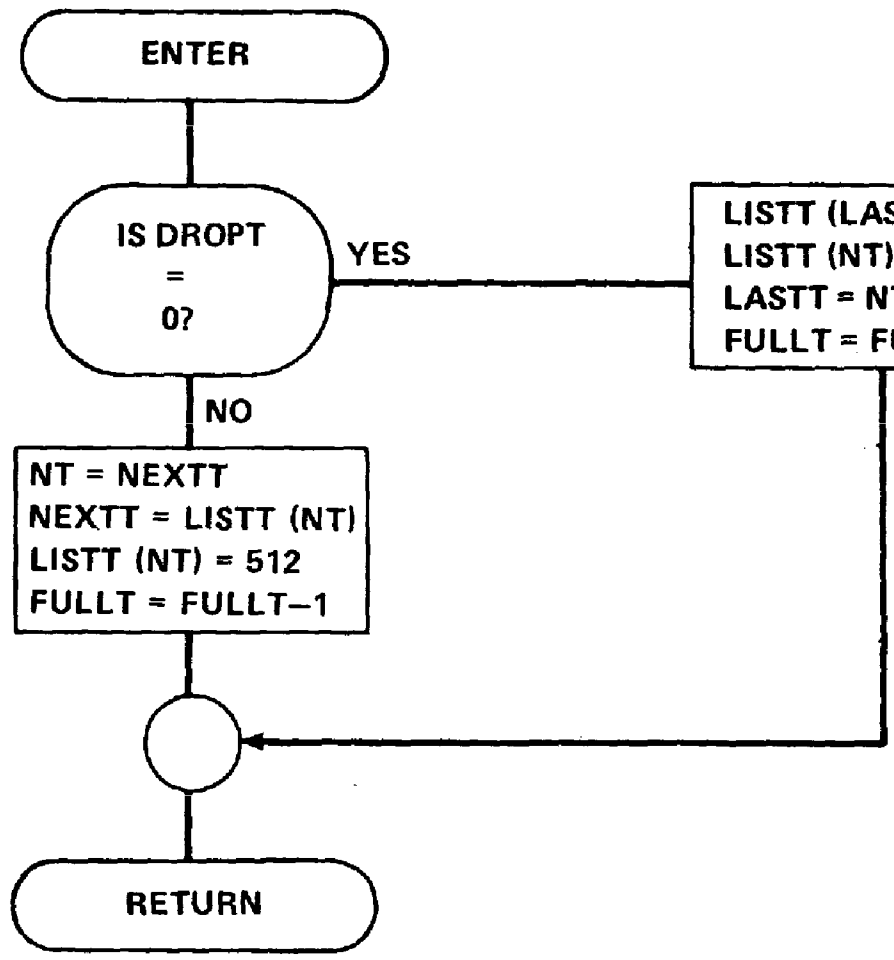


Fig. 2 — Flowchart for track number file, subroutine TRKNO (NT, DROPT)

When a new track number is requested, DROPT is set equal to one, and the system checks to see if FULLT is zero. If FULLT is not equal to zero the routine is called. Since DROPT = 1, the new track is assigned the next available track number; i.e., NT = NEXTT. The next available track number in the list is found, and NEXTT is set equal to LISTT (NT). FULLT is decremented, indicating that one less track number is available. Finally, LISTT (NT) is set equal to 512 (a number larger than the number of possible tracks). This is not necessary but helps in debugging the program.

A track number is dropped (DROPT = 0) by setting the last available track number LISTT (LASTT) equal to the track number NT, which is dropped. LISTT (NT) is set equal to zero to denote the last track number, and LASTT is then set equal to the track number being dropped, LASTT = NT. The parameter FULLT is incremented, indicating that one more track number is available.

The track and clutter number files maintain a linkage from one number to the next, and they operate very rapidly, eliminating searching techniques.

2.2 Track and Clutter Parameter Files

Parameters associated with a given track number are listed below. Parameter TF (NT)

Parameter	Description	
NR (NT)	Smoothed range stored every 8 scans of SPS-39	
RS (NT)	Smoothed range position	$x_s(k)$
AS (NT)	Smoothed azimuth position	
VRS (NT)	Smoothed range velocity	$v_s(k)$
VAS (NT)	Smoothed azimuth velocity	
RPT (NT)	Predicted range position	$x_p(k)$
APT (NT)	Predicted azimuth position	
ES (NT)	Elevation angle	
TT12 (NT)	Last time the SPS-12 updated the target	
TT39 (NT)	Last time the SPS-39 updated the target	
TT (NT)	Last time the target was updated	
TF (NT)	Time of targets first detection/Elevation scan parameter	
TTL12 (NT)	Next time the SPS-12 will see the target	
TTL39 (NT)	Next time the SPS-39 will see the target	
KT (NT)	0 firm track, 1 tentative track	
OUT (NT)	Output for display	
RPC (NC)	Point clutter's range	
APC (NC)	Point clutter's azimuth	
TC12 (NC)	Last time the SPS-12 updated the clutter	
TC39 (NC)	Last time the SPS-39 updated the clutter	

is used to store the time of the first detection until a firm track has been established. After a track has been established, it is used as a counter to determine on what scan of the SPS-39 an elevation scan will be performed on the target.

The parameter OUT (NT) is used for the display. Its format is listed as follows:

Bit	Condition
0	0 valid track; 1 invalid
1	1 if SPS-12 is detecting a target
2	1 if SPS-39 is detecting a target
3	1 if SPS-10 is detecting a target
4	1 if IFF is detecting a target
5	1 if the track is being handed off
6	1 if elevation information is requested.

2.3. Track Number Assignment to Azimuth Sector Files

The azimuth-range plane is separated into 64 equal azimuth sectors, each of 5.625°. After a track is updated or initiated, the predicted position of the target is checked to see which sector it occupies, and the track is assigned to this sector. If the track is dropped or moves to a new sector, it is dropped out of the sector in which it was previously located. The parameters associated with sector files are listed below. Only the assignment of track

Parameter	Description
TBX (*I*)	First track number in sector *I* (a subscript of array IDT)
IDT (256)	Each location corresponds to a track number, and the location contains the next track number in sector *I* or a zero.
CBX (*I*)	First clutter number in sector *I* (a subscript of array IDC)
IDC (256)	Each location corresponds to a clutter number, and the location contains the next clutter number in sector I or a zero.

numbers to azimuth cells is described, since the clutter number assignment is identical, and the process is essentially the same as described in Ref. 3. The TBX (*I*) file contains the first track number in sector *I*. If TBX (*I*) = 0, there are no tracks in sector *I*. The IDT (256) file has storage locations corresponding to each of the possible 256 track numbers. The first track number in sector *I* is obtained from FIRST = TBX (*I*). The second track number in the sector is obtained by NEXT1 = IDT (FIRST). The next track number in the sector is obtained by NEXT2 = IDT (NEXT1). This process is continued until a zero is encountered, indicating that there are no more track numbers in the sector.

When a new track is added or a track moves from one sector to another, a track number must be added to the sector. The flowchart for achieving this is shown in Fig. 3. The first track number in the sector is stored, the track number NT being added is made the first track number in the sector, and the track number NT in the IDT (NT) file is made equal to the original first track number in the sector. This procedure is essentially a push-down stack, pushing the older track numbers further down in the file.

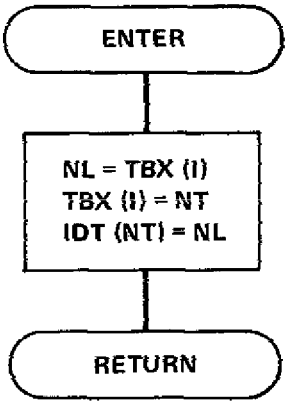


Fig. 3 — Adding a track number NT to sector I, subroutine TNEW (NT, I)

When a track is dropped or moves out of the sector, the track number must be removed from the sector. The flow diagram for this is shown in Fig. 4. First, it is determined whether the first track number in the sector TBX (I) is the one being dropped. If it is, the first track number in the sector is set equal to the second track number in the sector, and the location in the IDT file corresponding to the track number NT being dropped is set to zero. NT is set equal to the track number in the file following the one just dropped, so that we now have the next available track number. If the track number being dropped is not the first one in the file, then the push-down stack IDT (NL) is searched sequentially until the track number is found. The variable IDT (NL) containing NT as the next track number is replaced by the next track number following NT, and the variable in the IDT file corresponding to NT is set equal to zero. Again, NT is set equal to the track number in the file after the one being dropped.

The subroutines are given in Appendix B.

2.4. Input Data Bank

The basic input data from the radars can be broken into two categories; radar measurements and control parameters. The parameters associated with the input data are listed below. The input data to the SPS-12 will be discussed first. Two small buffers are

Input Parameters	Description
RM12 (K)	Range measurement off SPS-12, Kth detection
AM12 (K)	Azimuth measurement off SPS-12, Kth detection
TM12 (K)	Time of measurement off SPS-12, Kth detection
TAG12 (K)	Used in program (0 no correlation, 1 correlation)

(Con't)

Input Parameters	Description
RM39 (L)	Range measurement off SPS-39, Lth detection
AM39 (L)	Azimuth measurement off SPS-39, Lth detection
TM39 (L)	Time of measurement off SPS-39, Lth detection
TAG39 (L)	Used in program (0 no correlation, 1 correlation)
EM39 (L)	Elevation measurement off SPS-39, Lth detection
MRK12 (I)	Time the SPS-12 crosses the Ith sector
NP12 (I)	Position of pointer in buffer at Ith sector crossing
NB12 (I)	Number of detections in buffer in the Ith sector
P1239 (I)	Position of SPS-39 when SPS-12 crosses the Ith sector
I12T	Sector SPS-12 last crossed
MRK39 (J)	Time the SPS-39 crosses the Jth sector
NP39 (J)	Position of pointer in buffer at Jth sector crossing
NB12 (J)	Number of detections in buffer in the Jth sector
P3912 (J)	Position of SPS-12 when SPS-39 crosses the Jth sector
I39T	Sector SPS-39 last crossed.

external to the computer. On each range sweep of the radar one buffer is accepting data on detections, and the other is reading the data accumulated in it during the previous sweep. The buffers alternate on each range sweep of the radar. A binary counter that counts from $K = 0$ through 255 is used. Each time the data block RM12 (K), AM12 (K), TM12 (K) is read via a DMA (direct memory access) channel into the computer the counter is incremented by one. The counter total plus some prefixed constant represents the core location of each detection measurement in the computer. When the counter reaches 255 the next count goes to zero, and the counter is recycled. If the tracking system is working reasonably close behind the radar the data are never written over before being used. The timing and control parameters are discussed next.

There exist 64 equally spaced azimuth sectors of 5.625°, i.e., $I = 1$ through 64. As the radar crosses a sector boundary, five parameters associated with the Ith sector are read into the computer. The parameters are the time the SPS-12 crossed the Ith sector boundary MRK12 (I), the value of the binary counter used for addressing the input data NP12 (I), the number of target reports that occurred in the Ith sector NB12 (I), the position of the SPS-39 at the sector boundary P1239 (I), and the sector number I12T.

The data for the SPS-39 are read into the computer in the same manner but through a separate system. All data are read through a daisy-chain priority DMA channel with the SPS-12 sector information having top priority, followed by SPS-39 sector information, SPS-12 detection data, and SPS-39 detection data.

The clock used is a 15-bit binary counter that counts every 8 ms. The clock counts through approximately 4.4 min. before recycling. Nothing in the program is ever referenced beyond about 1 min in the past, and therefore clock recycling can easily be detected and compensated for.

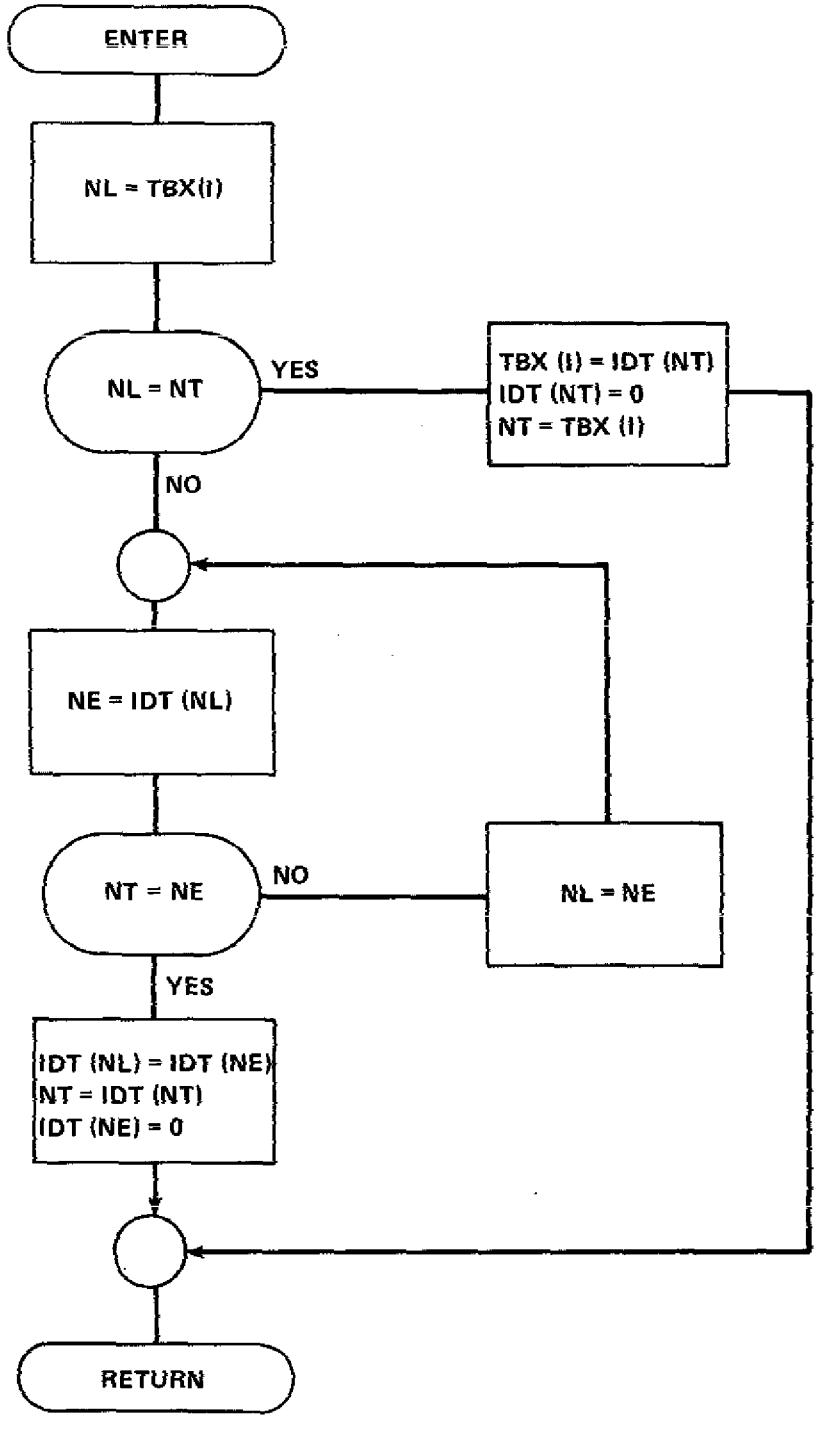


Fig. 4 — Dropping a track number NR from sector I, subroutine TDROP (NT, I)

The word TAG12 (K) or TAG39 (L) is used in the program to indicate whether the Kth detection from the SPS-12 or the Lth detection from the SPS-39 correlated with a track or not. It takes on value zero or one.

2.5. Modulo Arithmetic

The external clock just discussed recycles about every 4.4 min. Every time parameter in the program is referred to this clock. In addition, the azimuth recycles every 360°. In order to appropriately handle these conditions in the program, modulo arithmetic is used. Consider the addition of two numbers $A \oplus B$. If $A \oplus B$ is greater than the modulus, the modulus is subtracted from the sum. For a 15-bit representation of a number, A and B are both divided by two, then added using a modulus of 14 bits. The result is multiplied by two to achieve the 15-bit representation. This procedure is required to keep the machine from overflowing.

In the case of subtraction, $A \ominus B$, the result should be small relative to the modulus. A large result implies a wraparound problem. A large positive result requires the subtraction of the modulus, a large negative result requires the addition of the modulus. Since only positive values are subtracted, overflow is not a problem.

2.6. Other Parameters

Other parameters used in the program are listed in this section. These include display, elevation scan, status, program, and dummy parameters. Some of these parameters are self-explanatory. Others will be described more thoroughly in later sections.

The program-related parameters are as follows.

Parameter	Description
I12D	Sector in which tracks are presently being updated by SPS-12
I39D	Sector in which tracks are presently being updated by SPS-39
V12	Rotational velocity of SPS-12
V39	Rotational velocity of SPS-39
VRMIN	Range velocity for determination of target or clutter
VAMIN	Azimuth velocity for determination of target or clutter
TCMAX	Time a clutter is kept without an update
TTMAX	Time a track is kept without an update
TTLAG	TTMAX + ϵ for modulo clock
TNMAX	Time a tentative track is kept without an update
TFIX	Time after an initial detection before a decision is made

(Con't)

Parameter	Description
KONST	Large number for determining a correlation
CRC	Range correlation region size for clutters
CAC	Azimuth correlation region size for clutters
CRT (8, 2)	Range correlation region size for tracks*
CAT (8, 2, 16)	Azimuth correlation region size for tracks*
RALPA (128)	Range filter smoothing parameter α (time)
AALPA (128)	Azimuth filter smoothing parameter α (time)
RBETA (128)	Range filter smoothing parameter β (time)
ABETA (128)	Azimuth filter smoothing parameter β (time)

*Function of time since last update, whether tentative or firm track, and range (only in azimuth).

The status parameters are

Parameter*	Description
ISTA (1)	NTARGET, number of target tracks
ISTA (2)	Total number of tracks
ISTA (3)	Number of clutter points
ISTA (4)	NCATCH, number of times ALPNM is called per scan (proportional to free processing time)
ISTA (5)	I12DEL, present sector lag on SPS-12
ISTA (6)	NELEV, number of targets in elevation search
ISTA (7)	ISKIP; 1 indicates that sectors have been skipped on this scan.

*ISTA (8) to ISTA (12) are not presently used.

The alphanumeric parameters are listed here.

Parameter	Description
IOPER	Operational code specifying request
IPAR1	Parameter stating information about request
IPAR2	Parameter stating informat on about request
NUM	Number of targets that fulfill request
JTAR (32)	Track numbers that fulfill request
NHAND	Track number of target being handed off to the tracking radar
ISTART	Restart sector if NUM > 32

An elevation search is performed every four scans for the specified target. The elevation parameters are as follows.

Parameter	Description
NDESTAR (4)	Track numbers of targets designated by radar operator on present scan for elevation searches on the next scan
IAZIM (16)	Eight azimuth-range pairs designated to SPS-39 in order to perform elevation search
NTARPR (8)	Previously designated track numbers on which elevation searches will be performed on next scan
NCOUNT	Number of previously designated targets for which elevation searches are requested on the next scan

The following are dummy parameters often used in the system.

Parameter	Description
RM	Measured range of target corresponding to track or clutter
AM	Measured azimuth of target corresponding to track or clutter
TH	Measured time of detection corresponding to track or clutter
NDEL 1	Time difference
NDEL 2	Time difference
D	Distance from predicted position to nearest detection under correlation or equal to KONST
DI	Distance from predicted position to detection under correlation
DELR	Difference between predicted position and target report in range
DELA	Difference between predicted position and target report in azimuth
ISECT	Sector location
IFLIP	0 or 1, denoting correlation with firm or tentative tracks

The value of the least significant bit for certain parameters are

Parameters	Least Significant Bit
Range positions	31.25 ft
Azimuth positions	0.010986°
Range velocity	0.125 ft/s.
Azimuth velocity	0.000244 deg/s.
Time	0.008 s.

2.7. Smoothing Filter

A track is updated by computing a smoothed position and velocity. Then its position for the next update is computed. The filter used is an α - β filter [4]:

$$\begin{aligned} x_s\ (k) &= x_p\ (k) + \alpha[x_m(k) - x_p(k)] \\ v_s\ (k) &= v_s\ (k - 1) + \beta[x_m(k) - x_p(k)]/T_1 \end{aligned} \tag{1}$$

where

$x_p\ (k + 1) = x_s\ (k) + v_s(k)T_2$ (computed elsewhere) and

T_1 = DELT, time between current time and last update,

T_2 = time between current time and next update.

The internal parameter names for $x_s\ (k)$, $v_s\ (k)$, and $x_p\ (k + 1)$ are given on page 4 for both range and azimuth. The values of α and β are given as a function of time according to the Eq. [4].

$$\begin{aligned} \alpha &= 1 - e^{-2\zeta\omega_o T_1} \\ \beta &= 1 + e^{-2\zeta\omega_o T_1} - 2\, e^{-\zeta\omega_o T_1} \cos \omega_d T_1 \end{aligned} \tag{2}$$

where ζ and ω_o are constants, and ω_d is the damped natural frequency. The values α and β are stored in tables as a function of T_1, where T_1 is made an integer value using different stepping intervals for different regions of T_1. The parameters used in the routine for α and β are shown on pages 9 and 10, and the subroutine is shown in Appendix C along with the values of α and β used.

There are several precautionary notes. First, roundoff error and overflow conditions must be considered in the routine. Second, the azimuth wraparound problem must be handled.

2.8. Calculation of Time Until Next Update

Given that a track has just been updated (or an update has been attempted) with a detection from the SPS-12, subroutine TME12 (TH, T_2, NT) is called. The subroutine answers two questions. The first is, at what time does the SPS-12 see the target again? This is estimated by dividing an azimuth of 360° by the difference between the azimuth velocity of the radar V12 and the azimuth velocity of the target VAS (NT). This result is added to current time TH to give the desired result, TTL12 (NT). A similar calculation is made when a track is being updated with measurements from the SPS-39 in subroutine TME39 (TH, T_2, NT) to obtain TTL39 (NT).

The second question is, How much time T_2 will elapse between the current time TH and the time the next radar has the opportunity to detect the target? In TME12 (TH, T_2, NT) TTL12 (NT) has just been computed, and TTL39 (NT) is known. The current time is subtracted from both quantities, and the minimum difference is taken as T_2, defined as the increment in time between the time the target will next be updated and the current time. The current time TH is approximated by taking the time of the sector crossing MRK12 (I) for the sector in which the target lies. Similar calculations are made with TME39 (TH, T_2, NT). The routines are shown in Appendix D.

3.0. EXECUTIVE

The executive controls the basic timing of the program and interfaces the program with some of the external functions. The basic flowchart of the executive is shown in Fig. 5, and the routine is in Appendix E. When the system is turned on, files are initiated, and the system waits until the SPS-12 leads the SPS-39 by between 4 and 40 sectors. Then I12D is set equal to I12T, I39D is set equal to I39T, and the system jumps to the starting point in the executive. The valves of I12T and I12D run from 0 through 63 and represent the last sector crossing of the SPS-12 and the next sector to be updated on the SPS-12, respectively. If I12DEL = I12T θ I12D* is greater than 20 we say the system is overflowing (processing lagging too far behind the radar). In this case new update times for each track are computed in the sectors skipped, and the system returns to the beginning of the executive. This essentially ignores all the data in these sectors; the tracks are not updated. If I12DEL is less than 5 the program is said to be "caught up." Operator requests are accepted, and data are output to the alphanumeric display. The parameter I12DEL is a measure of the processing lag behind the radar. The processing lag is monitored only for the SPS-12 radar, since it is rotating faster than the SPS-39. The basic timing of the program will be examined next.

First, a simple example: Consider two radars rotating at identical speeds separated by 180° in azimuth. The ideal manner of processing would be to update simultaneously tracks separated by 180° with detections from each radar. Because of the sequential nature of computer processing, simultaneous updating is impossible. Therefore, tracks are alternately updated with detections from each radar. Now let the radars rotate at different speeds. If the sector to be next updated by detections from each radar is I12D for the SPS-12, and I39D for the SPS-39, the sector farthest back in time is updated first. When the two radars are near each other but not in the sector in which they cross, this method always keeps the detections that occurred first, updating the tracks before later detections are considered. In the crossing sector an inversion can occur; that is, later detections update the tracks before detections occurring earlier do. However, this inversion occurs over a very short time interval and in different sectors on a scan-to-scan basis. If a target is detected on both radars in the crossing sector, both detections are used to update the track. However, if the filter is carefully examined, the first detection processed, even if a time inversion has occurred, will be given significant weight. The second one will be given essentially zero weight. This does not adversely affect the track unless the time inversion is long, a situation that is impossible. If a detection occurs on one radar and not the other or none on either one, the track update is not affected.

*$\oplus, \theta$ denotes modulo arithmetic.

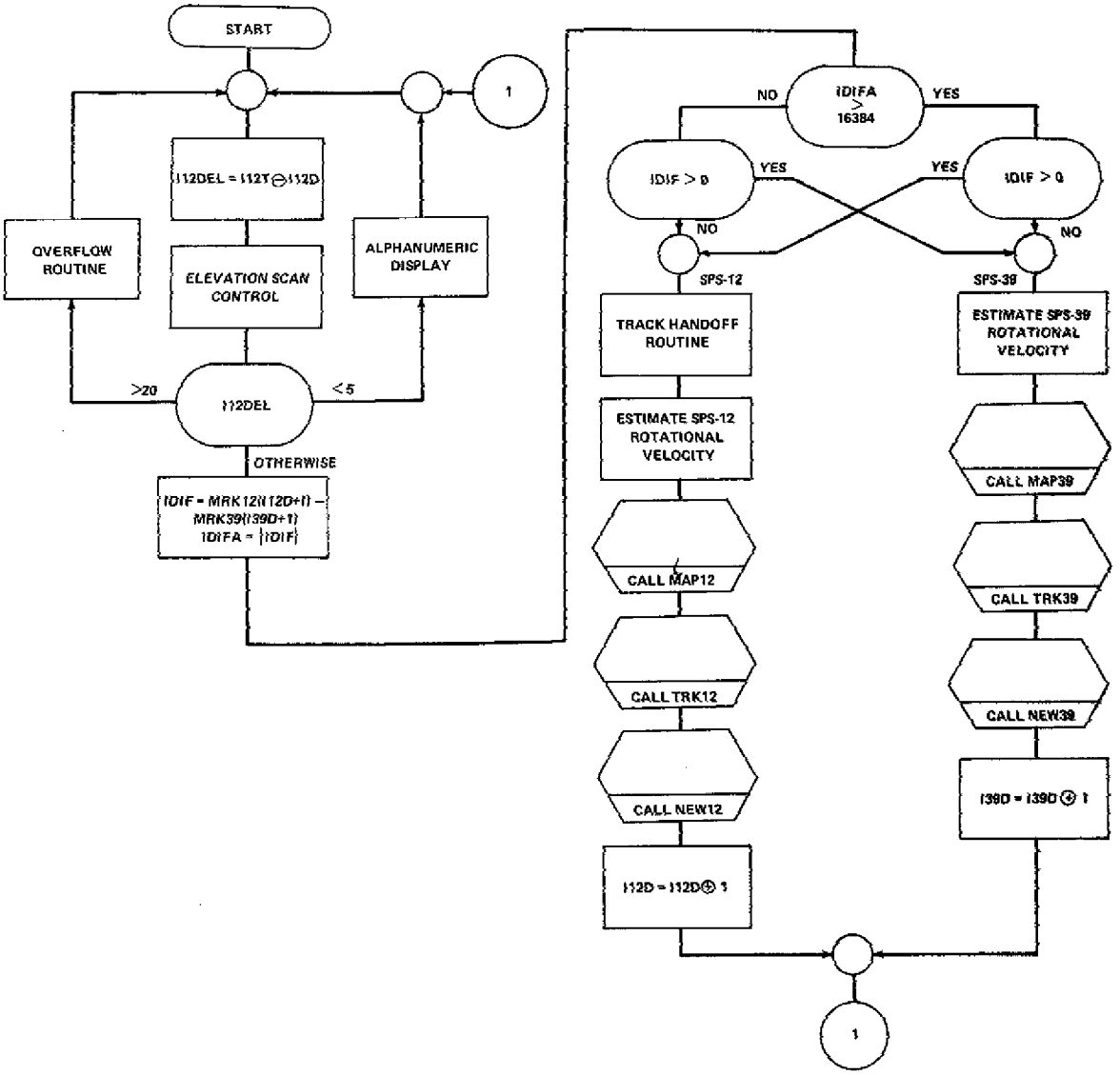


Fig. 5 — Flowchart of the executive, subroutine

From Fig. 5, the indices of MRK12 (.) and MRK39 (.) run from 1 to 64 while I12D and I39D run from 0 to 63. Therefore, it is necessary to add one to I12D and I39D when computing the time difference IDIF = MRK12 (I12D + 1) - MRK39 (I39D + 1). The recycling of the clock is neglected for the moment, and if IDIF > 0, detections from the SPS-39 are used to update tracks in sector I39D, since they occurred earlier than those in sector I12D of the SPS-12. The clock recycling problem is solved as follows. The sector crossing times MRK12 (I12D + 1) and MRK39 (I39D + 1) are fairly close together in real

time, and if the times are on opposite sides of zero on the clock, the magnitude of the difference is much larger than 16,384.* By considering the sign on IDIF we can determine which detections (SPS-12 or SPS-39) occurred earlier in time, in order to choose which radar will update the tracks next.

Once the radar has been chosen, the clutter routine is called. The clutter routine operates one sector in advance of the sector counter (I12D $\oplus$ 1 or I39D $\oplus$ 1) and removes from the detector file all detections that correlate with the point clutters. Then, in the tracking routine, the remaining detections are used to update first the target tracks and then the tentative tracks in the current sector (I12D or I39D). Finally in the track-initiating routine, all detections in the sector behind the sector counter (I12D θ 1 or I39D θ 1) that have not correlated with either point clutters, target tracks, or tentative tracks are used to initiate tentative tracks. The sector counter (I12D or I39D) is incremented, and the routine returns to the beginning of the executive to determine which sector will be updated next, and by which radar. In essence, the executive closely updates tracks with detections occurring sequentially in time.

4.0. CLUTTER MAP

The clutter map, subroutine MAP12, removes from the radar detections associated with point clutters or slowly moving targets. The flowchart for the clutter map using detections from the SPS-12 is shown in Fig. 6.

The clutter map operates one sector in advance of the sector location I12D (where tracks are to be updated), and all detections from the SPS-12 associated with clutter points are removed before any tracks are updated. The clutter numbers in the clutter sector files are called up one by one to be updated. The following time differences are calculated for each clutter point: time since last update by SPS-12, given by MRK12 (I12D + 1) - TC12 (NC), and time since last update by SPS-39, given by MRK12 (I12D + 1) - TC39 (NC). If both the time differences exceed 40 s, the point clutter is dropped from the clutter number and sector files, and the next clutter number in the sector is obtained. If the clutter point has been updated by the SPS-12 within the last second, then it has been updated by the SPS-12 on the current scan of the radar and is being considered again because it has changed sectors. In this case, the clutter point is ignored and the next clutter point is obtained. If, as is the usual case, the clutter point has been recently updated, but not on the current scan of the SPS-12, the clutter number is presented to the correlation part of the clutter map.

The correlator attempts to correlate each clutter point in sector I12D $\oplus$ 1 with all detections in sectors I12D, I12D $\oplus$ 1, and I12D $\oplus$ 2. A detection is said to correlate with a clutter point if the distance DELR, which is the difference between the range to the clutter point and the range to the detector, is less than some distance CRC and if the angle DELA, which is the difference between the azimuth angle of the clutter point and the azimuth angle of the detection, is less than some angle CAC. If the detection does

*The clock recycles every 32,768 counts (2^{15}).

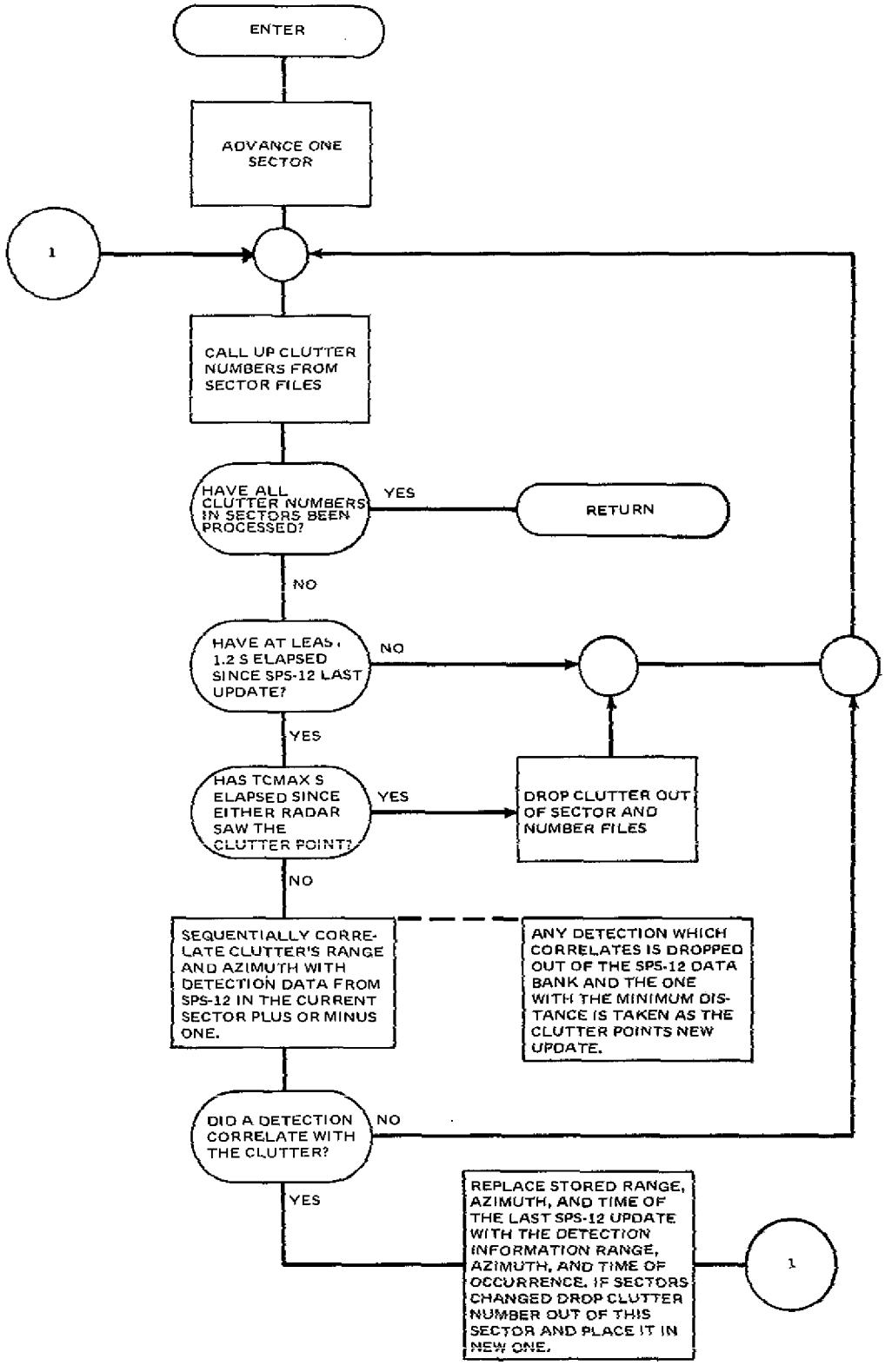


Fig. 6 — Flowchart for clutter map for detections, subroutine MAP12 sub TRK12

not correlate with the clutter point, the next detection is examined. If the detection does correlate, the effective distance DI of the detection from the clutter point is calculated from

$$DI = \left(\frac{DELR}{CRC}\right)^2 + \left(\frac{DELA}{CAC}\right)^2$$

The detection that correlates with clutter and has minimum effective distance is the one chosen to update the clutter point.

Any detection that correlates with a clutter point is removed from the input buffer file as follows: The location of the parameters of the first detection in a sector I is contained in NP12 (I), and the number of detection is contained in NB12 (I). A detection is removed from the sector by replacing its parameters of range, azimuth, and time by the parameters of the last valid detection in the sector, and decrementing the contents of NB12 (I) by one. In this manner all good detections in a sector are listed sequentially from the core locations of the first detection in the sector through the number of good detections left.

When a detection updates a clutter, the measured positions and time of the detection are stored in RPC (NC), APC (NC), and TC12 (NC). If the clutter changes sectors, the clutter number is dropped out of the current clutter sector file and reinserted in the correct sector file. Whether a clutter point is updated or not, after all good detections have been examined, the next clutter number is obtained for processing. After all clutter points lying in this sector I12D ⊕ 1 have been processed, the routine attempts to update tracks in sector I12D. The flow diagram for MAP39 is identical except that detections from the SPS-39 are used to update the clutter points.

In summary, the clutter file stores the locations of the point clutters or slowly moving targets and removes the detection from the radars associated with them. If too long a time has elapsed between updates the clutter point is dropped. The subroutines MAP12 and MAP39 are shown in Appendix F. Note the handling of the wraparound problem with the clock and in azimuth.

5.0. TRACK UPDATING

The tracks are updated in routines TRK12 and TRK39. The flow diagram for track updating by using detections from the SPS-12 is shown in Fig. 7.

A track number is obtained from the track sector files, and a filtering process takes place. First, all tentative tracks [KT (NT) = 1] are skipped until all target tracks [KT (NT) = 0] are processed; the tentative tracks are processed on a second pass through the track sector files. Second, all tracks whose predicted time of updating [TTL12 (NT)] is more than 1.2 away from the time of the sector crossing [MRK12 (I12D)] are skipped, on the assumption that they have just been updated on the current scan and have been obtained for processing because they have changed sectors. The track is now ready for

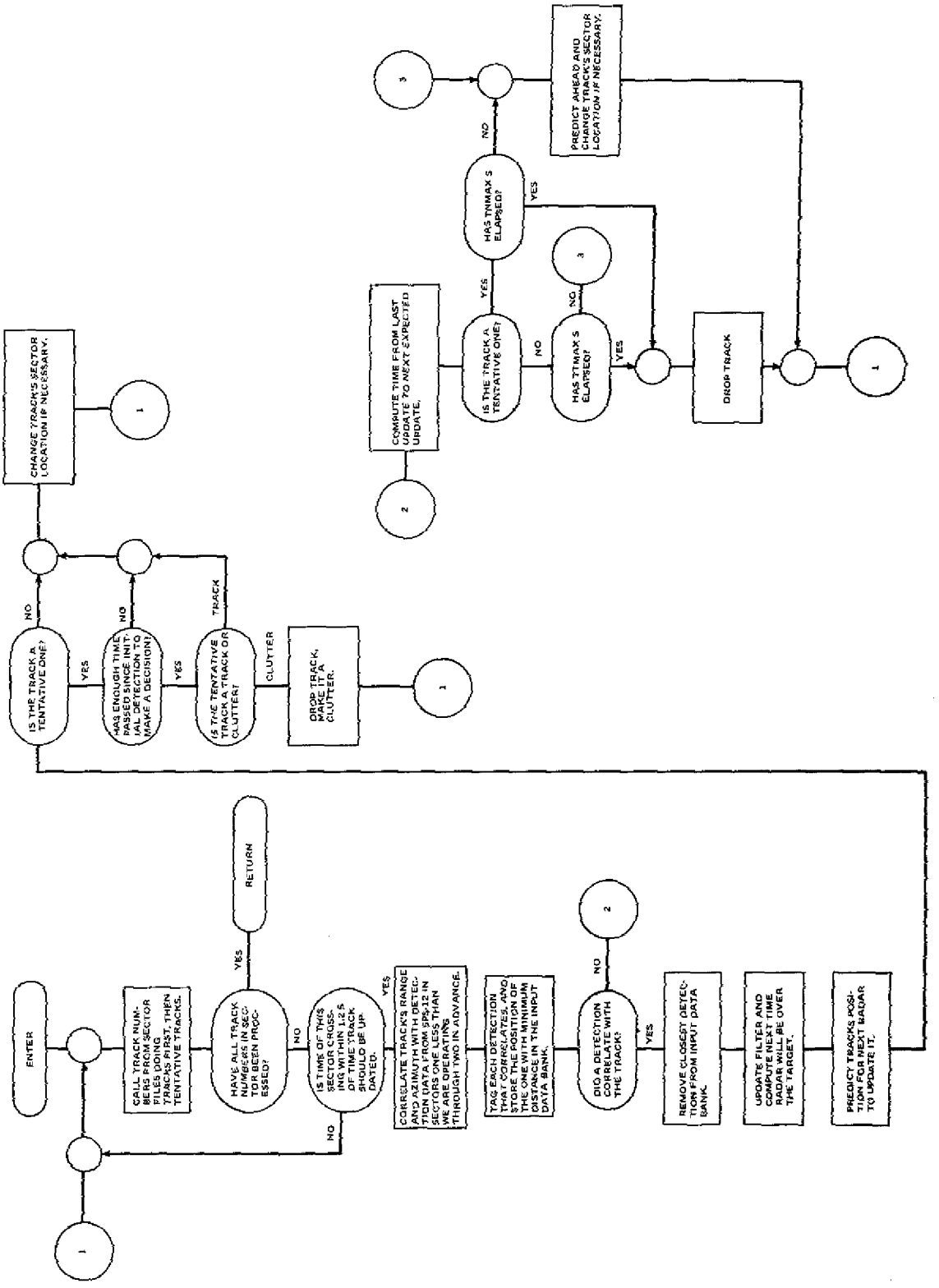


Fig. 7 — Routine for updating tracks

correlation. The correlator is similar to the clutter correlator described previously. The correlator attempts to correlate each track in sector I12D with all detections remaining in sectors I12D θ 1, I12D, I12D $\oplus$ 1, and I12D $\oplus$ 2. A detection is said to correlate with a track if the distance DELR (|predicted track range minus detection range|) is less than the distance CRT (., .), and if the angle DELA (|predicted track azimuth minus detection azimuth|) is less than CAT (., ., .). CRT (., .) is a function of the time since last update of this track and whether the track is tentative or a target, and CAT (., ., .) is a function of time, whether the track is tentative or a target, and the range to the predicted track. If the detection does not correlate with the track, the next detection is examined. If the detection does correlate with the track, the effective distance DI of the detection is calculated from

$$DI = \left(\frac{DELR}{CRT\ (.,\ .)}\right)^2 + \left(\frac{DELA}{CAT\ (.,\ .,\ .)}\right)^2.$$

Each detection that correlates with a track is indicated by setting TAG12 (.) equal to one. After all detections have been processed with a track, the detection with the minimum effective distance is used to update the track, and that detection is dropped from the detection files. Any other detections that correlate with the track remain in the detection files to update other tracks but cannot be used to initiate new tracks. When a track is updated, subroutine FILTR (Sec. 2.7 and Appendix C) is entered, and the smoothed track parameters are generated. Then subroutine TME12 (Sec. 2.8 and Appendix D) is entered, and the elapsed time until the next opportunity to update this track is calculated. This time is used in conjunction with the smoothed position parameters and velocities to generate the predicted positions of the track at the next opportunity to update. If the motion of the track places the track in another sector, the track number is removed from the current sector of the track files and is inserted in the appropriate sector. If the track being updated is tentative, the elasped time since initiation of the track is checked. If this time is less than 16 s, the track is checked for a sector change, and the next track is obtained.

If the time is more than 16 s, the filter has had time to settle down, and the smoothed velocity is assumed accurate enough to differentiate between target and clutter points. If the smoothed velocity is low enough (range velocity of 60 knots, azimuth velocity of 0.2 deg/s) the tentative track is placed in the clutter map (if unused clutter numbers exist). Fast-moving tentative tracks are confirmed as target tracks by setting KT (NT) equal to zero. Again the predicted position is checked for a sector change, and the next track is obtained for processing.

If a track does not correlate with any detection, the disposition of the track depends on the type of track and the time since it was last updated. The time of the next opportunity to update is calculated. If the difference between this time and the time of the last update is greater than 17 s for a tentative track or 40 s for a target track, the track is dropped by removing the track number from the track number files. If the time since the last update is small the track is "coasted" i.e., the predicted position at the time of the next update is calculated using the smoothed parameters at the time of the last update. The track is checked for a sector change, and the next track number is obtained. The tracking part of the program is terminated after all tracks in the sector have been processed.

The routine TRK39 is almost identical to TRK12. It uses detections from the SPS-39 rather than the SPS-12. No filter is used in elevation, and the last measured elevation is stored. Every eight scans the routine checks the range velocity of a target track by using the current range and the range eight scans previously and decides if the target track is a point clutter. The display information is set in TRK39.

In summary, tracks are designated as target tracks or tentative tracks. No track is allowed to be updated by the same radar more than once in a scan. No detection is allowed to update more than one track, and any detection which correlates with any track is not allowed to initiate a new track. When a track is associated with a detection, the track is updated and its position is predicted ahead to the time of the next opportunity to update the track. When a detection does not correlate with the track, the track is not updated; however, its position is still predicted ahead to the time of the next opportunity to update. The subroutines TRK12 and TRK39 are shown in Appendix G.

6.0. TRACK INITIATION

After all the clutter points and tracks have been updated with the detections, the remaining detections are used to initiate new tracks. The flow diagram for initiating tracks from the detections from the SPS-12 radar is shown in Fig. 8.

Tracks are initiated one sector behind the sector in which tracks are updated. This ensures that all tracks have been updated before the remaining detections are used for initiating tracks. Each detection remaining in this sector is obtained from the input data bank. The detection is checked to see if it has ever been correlated with a track by looking at TAG12 (.). If it has, no action is taken. If it has not, a track is initiated, if track numbers are available.

A track is initiated by assigning a track number, setting the predicted and smoothed positions equal to the detection positions, and setting the velocities equal to zero. The track is made a tentative track and placed in the sector it was detected in. All times except the ones described below are set equal to the time of detection. The value of TT39 (NT) is set TTLAG seconds behind the time of detection to indicate that the SPS-39 radar has not yet updated the track. The times TTL12 (NT) and TTL39 (NT), which denote the next time each radar will see the target, are set.

In summary, one sector behind the sector of track updates, NEW12 initiates tracks for detections which have never been correlated. NEW39 performs the same operation with detections left from SPS-39 radar. The subroutines NEW12 and NEW39 are shown in Appendix H.

7.0. ALPHANUMERIC DISPLAY

Data requests for the alphanumeric display are processed by subroutine ALPNM. This routine is called by EXCUT on an available-time basis, i.e., when all available tracks four sectors behind the radar have been processed. The general operation of ALPNM is described in Sec. 7.1, and the specific operator requests are described in Sec. 7.2.

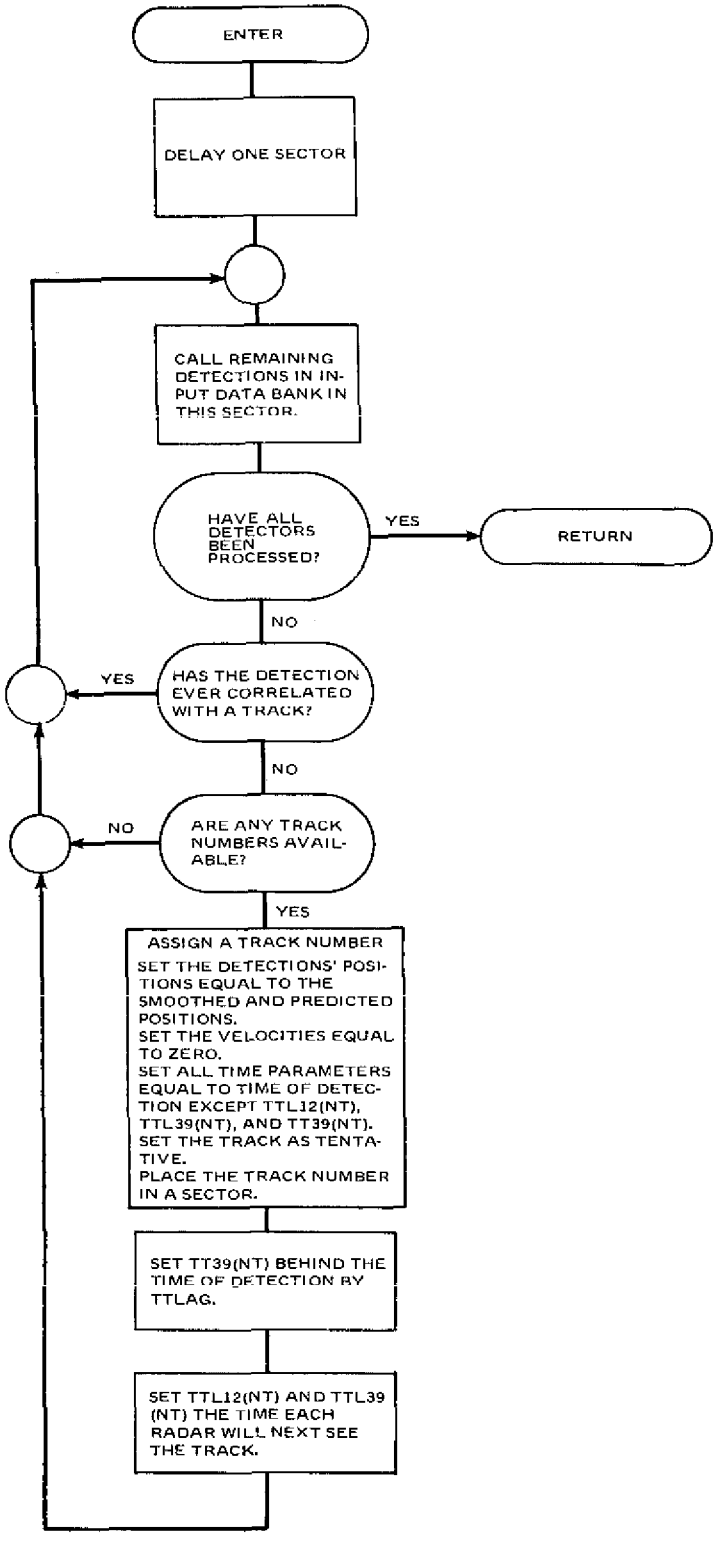


Fig. 8 — Initiation of new tracks, subroutine NEW12

7.1. General Operation

The radar operator enters three parameters (IOPER, IPAR1, and IPAR2) via a DMA channel. IOPER is presently a number between 0 and 7 specifying a request, and IPAR1 and IPAR2 are parameters stating information about the request. After the request has been processed IOPER is set to -1, NUM is the number of data words that fulfull the request, and JTAR (32) represents the output data words. JTAR contains either six target parameters for a specified target or the track numbers that fulfill a request. It should be noted that inside the computer, track numbers run 1 to 256, whereas on the outside (display) they run 0 to 255. If more than 32 numbers fulfill a request, NUM is set to 255 and ISTART is set to an appropriate value so that the remaining track numbers can be given when the operator repeats his request.

Approximately every half second the display equipment interrogates the location IOPER to see if it has been reset to -1, signifying completion. When a -1 is found, the appropriate values, NUM and JTAR (.), are read via a DMA. The detailed operating of the display equipment will be presented in a forthcoming report.

7.2. Operator Requests

The appropriate parameters for the different operator requests are given in the following list.

IOPER = 0 Target handoff
IPAR1 = track number
IPAR2 = 1 means end handoff request

IOPER = 1 List targets within an azimuth interval
IPAR1 = first azimuth
IPAR2 = second azimuth

IOPER = 2 List targets inside or outside a designated range
IPAR1 = range
IPAR2 = 1 (inside) or 2 (outside)

IOPER = 3 List target parameters
IPAR1 = track number

IOPER = 4 List all targets

IOPER = 5 List tentative tracks

IOPER = 6 List high-closing-velocity targets
IPAR1 = velocity

IOPER = 7 List targets under elevation search

The output parameters for IOPER = 0 or 3 are

NUM = 6
JTAR (1) = smooth range
JTAR (2) = smooth azimuth
JTAR (3) = elevation
JTAR (4) = last update time
JTAR (5) = smooth range velocity
JTAR (6) = smooth azimuth velocity.

The output parameters for all other requests are

NUM = number of tracks
JTAR (1) = track number
. . .
. . .
. . .
JTAR (NUM) = track number.

All searches are performed by searching through all sectors using the track map TBX (.) and the track indicator IDT (.).

8.0. ELEVATION SEARCHES

Azimuth angles at which the SPS-39 will perform elevation scans are calculated by subroutine ELEV. Subroutine ELEV is called by EXCUT once per scan of the SPS-39, approximately when the SPS-39 crosses its 61st azimuth sector. When the SPS-39 goes through 0°, azimuth-range pairs for elevation scans are written via a DMA from the computer into a shift register. The radar azimuth converter is compared to the designated azimuth. When the two are equal, four elevation scans are performed (this covers approximately 7° of azimuth), and detections are made in a range interval centered at the designated range. There can be as many as eight designated azimuths per scan.

8.1. Designated Targets

Elevation scans are performed on two types of targets: those the operator has just designated and those that have been previously designated. The operator either designates new targets or drops old targets by entering coded track numbers into NDESTAR (4). The eight least significant bits represent the track number, and the ninth bit is 1 if a target is newly designated and 0 if a target is to be dropped. If a target number NT is designated, NT is stored in NTEMP (.), and TF (NT) is set to 4. NTEMP (.) is a storage area for the track numbers on which elevation scans are to be performed, and TF (.) is a counter that indicates how many SPS-39 scans occur before the next elevation scan on this target. Every time one attempts to update a target in TRK39, TF (.) is decreased by

1 until it equals 1. When TF (NT) equals 1, TF (NT) is set back to 4, NCOUNT is increased by 1, and NT is stored in NTARPR (NCOUNT), which is an array of targets previously designated.

As an example, assume that there are INUM targets newly designated and NCOUNT targets previously designated. If (INUM + NCOUNT) $\leq$ 8, all track numbers are stored into NTEMP and NCOUNT is set to 0. If (INUM + NCOUNT) > 8, the INUM newly designated and the first (8 - INUM) previously designated targets are stored in NTEMP (.). NCOUNT is reset to INUM + NCOUNT - 8, and the unused previously designated targets are stored in the first NCOUNT locations of NTARPR (.). Then the update times on the next scan of the SPS-39 are found for each target; these times are used to calculate the predicted positions of the targets using Eq. (1). Approximately 3° is subtracted from all the predicted azimuths. These new azimuths now represent the angles where the elevation scans will begin. The azimuths are next ordered and the range-azimuth pairs for each target are stored in consecutive locations of IAZIM (.). If fewer than eight targets are designated on a scan, the rest of the array is filled with zeros. This array is used by the SPS-39 to perform the desired elevation scan. The detailed operation of the hardware will be described in a separate report.

8.2. Status Parameters

Since subroutine ELEV is called once and only once per scan, this routine provides a convenient place for setting the status parameters. These parameters are given on page 10 and are self-explanatory.

9.0. MONTE CARLO SIMULATION

To test the tracking logic and to obtain an estimate of the processing time, a computer simulation was written to generate radar data, i.e., range, azimuth, and time estimates for clutter points and targets. The simulation that generates radar data on tape is described in Secs. 9.1 to 9.4, the results of operating the tracking logic against two sets of radar data are given in Sec. 9.5, and various conclusions are given in Sec. 9.6.

9.1. Generation of Clutter Points

There are N_c clutter points, distributed uniformly in azimuth. Ninety percent of the clutter points lie between 5 and 32 n.mi., and the remainder lie between 32 and 92 n.mi. In each interval, the range is uniformly distributed. Eighty percent of the clutter points can be detected by both radars, 10% can be detected only by the SPS-39.

Mathematically, the statistical nature of the clutter points can be generated using a uniform random-number generator that generates a number U between 0 and 1. In the following discussion each U will represent a new random number. The azimuth of the target in degrees is

$$A_i = 360\ U_{i,1}. \tag{3}$$

The range in nautical miles is

$$R_i = \begin{cases} 30\ U_{i,2} + 2 & U_{i,2} > 0.1 \\ \\ 600\ U_{i,2} + 32 & U_{i,2} < 0.1\ . \end{cases} \tag{4}$$

Finally, let $C_{i,12}$ and $C_{i,39}$ indicate whether the SPS-12 and SPS-39 radars, respectively, can detect the clutter points. That is,

$$\begin{aligned} C_{i,12} = 1 \text{ and } C_{i,39} = 1 \quad &\text{if} \quad 0.0 \leqslant U_{i,3} < 0.8 \\ C_{i,12} = 1 \text{ and } C_{i,39} = 0 \quad &\text{if} \quad 0.8 \leqslant U_{i,3} < 0.9 \\ C_{i,12} = 0 \text{ and } C_{i,39} = 1 \quad &\text{if} \quad 0.9 \leqslant U_{i,3} \leqslant 1.0 \end{aligned} \tag{5}$$

where 1 is a detection and 0 is not a detection.

9.2. Generation of Targets

There are N_T targets, distributed uniformly inside a circle whose radius is $R_{\max}$. Eighty percent of the targets can be detected by both radars, 10% only by the SPS-12, and 10% only by the SPS-39. The speed of the targets is uniformly distributed between 500 and 1500 ft/s, and the heading is uniformly distributed in 360°

Mathematically, the target's initial coordinates are

$$\begin{aligned} x_i &= (1 - 2U_{i,1})\ R_{\max} \\ y_i &= (1 - 2U_{i,2})\ R_{\max} \end{aligned} \tag{6}$$

when

$$R_{\max}^2 \geqslant x_i^2 + y_i^2. \tag{7}$$

If Eq. (7) is violated, new $U_{i,1}$ and $U_{i,2}$ are chosen so that Eq. (7) is true. $T_{i,12}$ and $T_{i,39}$ are equivalent to $C_{i,12}$ and $C_{i,39}$ and are given by an expression similar to Eq. (5). The x and y velocities are given by

$$\begin{aligned} VX_i &= V_i \cos(\theta_i) \\ VY_i &= V_i \sin(\theta_i) \end{aligned} \tag{8}$$

where

$$V_i = 500 + 1000\ U_{i,3} \tag{9}$$

$$\theta_i = 360\ U_{i,4} \tag{10}$$

On every new scan of the SPS-39, the range of each target is calculated. If it exceeds R_{max}, it is replaced by a target with the following parameters:

$$\begin{aligned} x &= R \cos \alpha \\ y &= R \sin \alpha \end{aligned} \tag{11}$$

where

$$\begin{aligned} R &= 75 + 30\ U_1 \text{ (in nautical miles)} \\ \alpha &= 360\ U_2 \text{ (in degrees);} \end{aligned} \tag{12}$$

and

$$\begin{aligned} VX &= V \cos \theta \\ VY &= V \sin \theta \end{aligned} \tag{13}$$

where

$$\begin{aligned} V &= 500 + 1000\ U_3 \\ \theta &= \alpha + 180^\circ + 14(1 - 2U_4). \end{aligned} \tag{14}$$

These parameters specify a target entering the detection region.

9.3. Initialization of Times and Radars

The scan times of the two radars are randomized, so that

$$\begin{aligned} S_{12} &= 5.8 + 0.4U_1 \\ S_{39} &= 7.8 + 0.4U_2. \end{aligned} \tag{15}$$

The radars consequently have asynchronous rotation rates. The initial positions of both radars are randomly set to one of 64 sector crossings. The times and angles are defined as follows:

T is the present time
T_{12} is the time of the next sector crossing of the SPS-12

T_{39} is the time of the next sector crossing of the SPS-39
θ_{12} is the present position of the SPS-12
θ_{39} is the present position of the SPS-39.

Initially,

$$T = 0$$
$$T_{12} = S_{12}/64$$
$$T_{39} = S_{39}/64 \tag{16}$$
$$\theta_{12} = 360\ K_{12}/64$$
$$\theta_{39} = 360\ K_{39}/64$$

where K_{12} and K_{39} are integers uniformly distributed between 0 and 63.

9.4. Generation of Data

Data are generated sector by sector. First, time is incremented by

$$\Delta = \text{minimum } (T_{12}, T_{39}) - T \tag{17}$$

so that the present time is

$$T = \text{minimum } (T_{12}, T_{39}), \tag{18}$$

and the radar positions are $\theta_{12} + 360\Delta/S_{12}$ and $\theta_{39} + 360\Delta/S_{39}$. The target positions are also updated so that the new positions are $X_i + \Delta(VX_i)$ and $Y_i + \Delta(VY_i)$.

Without loss of generality, let us assume that $T = T12$. That is, radar detections will be generated for the SPS-12 in the azimuth interval

$$\theta_{12} - 1.5 \text{ to } \theta_{12} - 1.5 - 360/64. \tag{19}$$

The 1.5° corresponds to the lag associated with threshold crossing procedures for estimating azimuth position [1].

9.4.1. Detection of Clutter Points — First of all, the computer program searches through the clutter file to find all the clutter points in the interval defined by Eq. (17). For each point in the interval, $C_{i,12}$ is examined, and if the SPS-12 can detect the point, a random number U is generated and compared to P_c, the probability of detecting a clutter point. If $U > P_c$, the clutter point is not detected on this scan, and the next point is processed. On the other hand, if $U \leqslant P_c$, the clutter point is detected; and range, azimuth, and time measurements are generated. The range measurement is generated

by adding a Gaussian random variable, whose standard deviation is 0.3 of a range resolution cell $C_\tau/2$ to the range of the clutter point and then quantizing the result into an integral number of range resolution cells. That is

$$R_M = \frac{C_\tau}{2}\left[\text{Max Int}\left(\frac{2R_i}{C_\tau} + 0.3\ a_i\right)\right], \tag{20}$$

where a_i is a Gaussian random variable with mean 0 and variance 1. Since $C_\tau/2$ = 500 ft for the SPS-12, and 1 binary in the computer corresponds to 31.25 ft, 16 × R_M would correspond to the measured range that would be transferred to the tracking comptuer. In a similar manner, the measured azimuth is

$$\theta_M = \theta_i + 0.3a_i \tag{21}$$

and the azimuth sent to the computer is

$$8\ \text{Max Int}\ [2^{12}\theta_M/360] \tag{22}$$

where a_i is again a gaussian random variable with mean 0 and variance 1, and 360° corresponds to 2^{15}. The time is given by

$$T_M = T - \frac{(\theta_{12} - \theta_M)S_{12}}{360} \tag{23}$$

and the time transferred to the computer is

$$\text{Max Int}\ [T_M/0.008]\ \ \text{modulo}\ 2^{15} \tag{24}$$

where 1 bit of time corresponds to 8 ms.

9.4.2. Detection of Targets — The generation of target data is similar to the generation of clutter data except for two minor differences:

1. If the target range is less than 5 n.mi. or greater than R_{max}, the target is not detected.

2. The random number U is compared to P_T, the probability of detecting a target, instead of to P_c.

9.4.3. Bookkeeping Data — At the SPS-12 sector crossing, the following five bookkeeping variables are also calculated:

1. Radar: 12

2. Present sector: Max Int $[64\theta_{12}/360]$

3. Time: Max Int $[T/0.008]$ modulo 2^{15}

4. Position of other radar: 8 Max Int $[2^{12}\theta_{39}/360]$

5. Number of detections: N_D.

Item 5 is used to generate NP12 and NB12, and item 1 is used to indicate which radar is generating detections. In the actual system item 1 is not required, since the DMAs store the data from the two radars at different locations in core memory.

9.5. Monte Carlo Results

The tracking algorithm was programed in Fortran on the CDC 3800 computer and the Nova 800 minicomputer. The CDC 3800 was used to check the logic and to obtain a very accurate timing of the system. Two cases were examined: a low-target-density case and a medium-target-density case.

9.5.1. Low Target Density — In this simulation the following parameters were used:

N_c = 40 clutter points
N_T = 10 targets
P_c = 0.95 probability of detecting clutters
P_T = 0.90 probability of detecting targets
R_{max} = 106 n.mi.

In Fig. 9, the number of tentative tracks, target tracks, and clutters per scan of the SPS-12 are shown. On the first quarter of the SP-12's scan (a scan is counted each time the radar rotates through 0°), 12 tentative tracks are established. After the first full scan, 56 tentative tracks are established. More than 50 tracks (40 clutters plus 10 targets) are established, because in crossing sectors (the SPS-12 passing the SPS-39) two tracks are established on new detections. This problem can be avoided by choosing the starting time so that the radars do not cross on the initial scan. Since 16 s is the earliest that tentative tracks can be changed into clutters or targets, the number of tentative tracks is not reduced until the fourth scan. Then the number of tentative tracks is reduced sharply. By the sixth scan, 30 clutter points have been correctly identified as clutter, and 9 targets have been correctly identified. On the other hand, one target has been identified as clutter, and nine bogus tracks have been established. The incorrectly identified target is at a range of 97 n.mi., traveling at a speed of 700 ft/s. However, since the target is traveling perpendicular to the radial vector, its radial speed is only 10 ft/s, and its azimuthal speed is 0.06 deg/s. Thus, because of its low apparent speed the target is placed in the clutter file. The bogus tracks are caused by the fluctuating point clutters. During the tentative track phase, the correlation regions are large so that high-speed targets can be tracked. Consequently, when a point clutter fades and another one appears, a correlation is made, a high velocity is developed, and the track is declared a target. These bogus tracks either are not updated and are dropped after 40 s (for instance, two targets are dropped on scan 11), or are updated by a clutter point and become stationary. The latter tracks are removed by comparing the smoothed range every eight scans of the

SPS-39. If the target has moved less than 3000 ft, it is transferred to the clutter file. According to Fig. 9, targets are transferred on scans 9, 10, and 20 of the SPS-12. Finally, on scan 21, a target goes beyond R_{max}, and a new target enters the area. This new target is declared a target on scan 24, and the old target is dropped on scan 26.

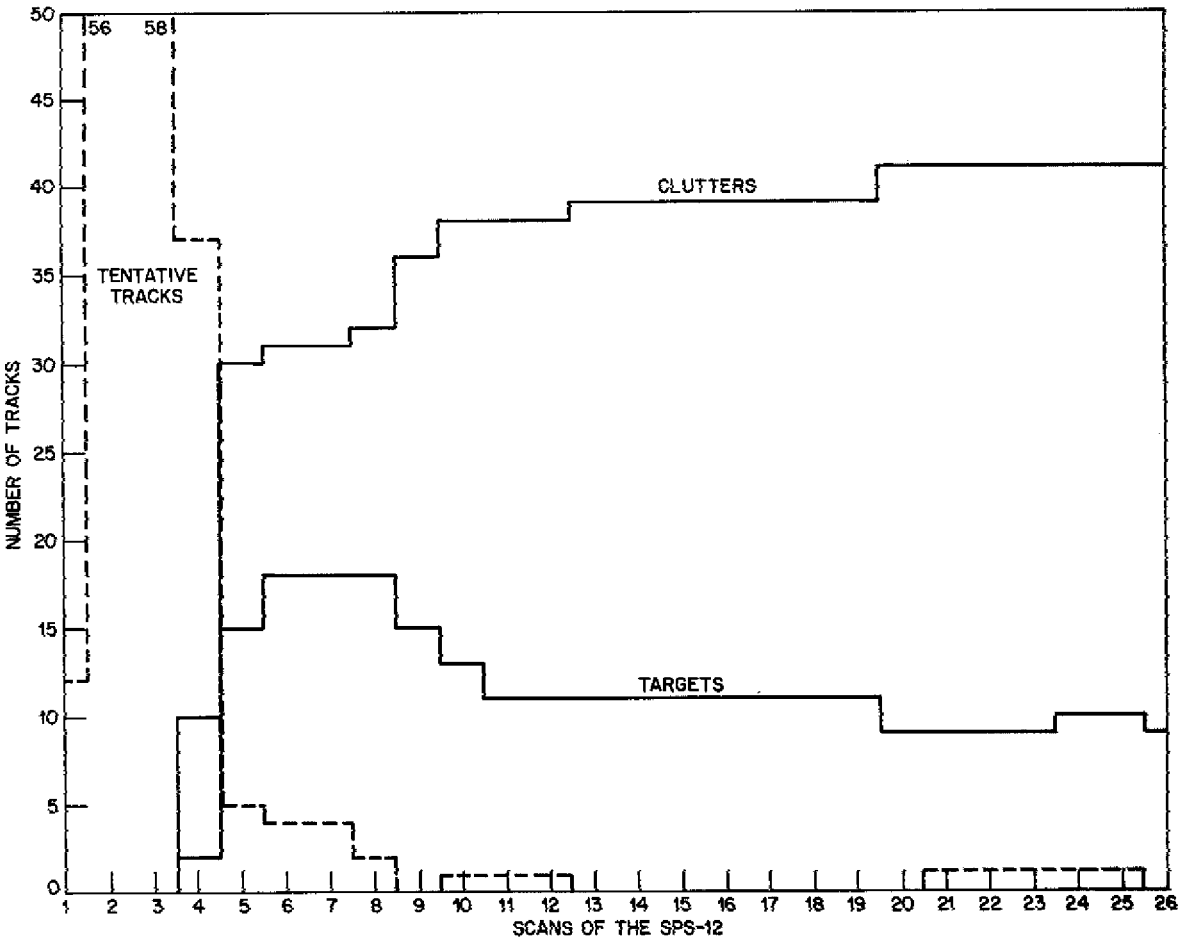


Fig. 9 — Number of tracks (clutter points, targets, and tentative tracks) vs scans of the SPS-12; low target density

The computation time on the CDC 3800 is shown in Fig. 10. The times were measured very accurately with the CDC 3800 time function. The startup times, scans 3 to 5, are large because all detections are at first assumed to be targets and consequently require track computations, which are more time-consuming than clutter updates. The steady-state computation time is approximately 140 ms.

The computation time on the Nova 800 is more difficult to obtain, since the real-time clock counts only in seconds. Consequently, the total computation time was found for 16 scans, and the computation time was assumed to be proportional to the computation

time of the CDC 3800 on a scan basis. Specifically, the Nova required approximately 21 s to process 16 scans and is 8.4 times slower than the CDC 3800.

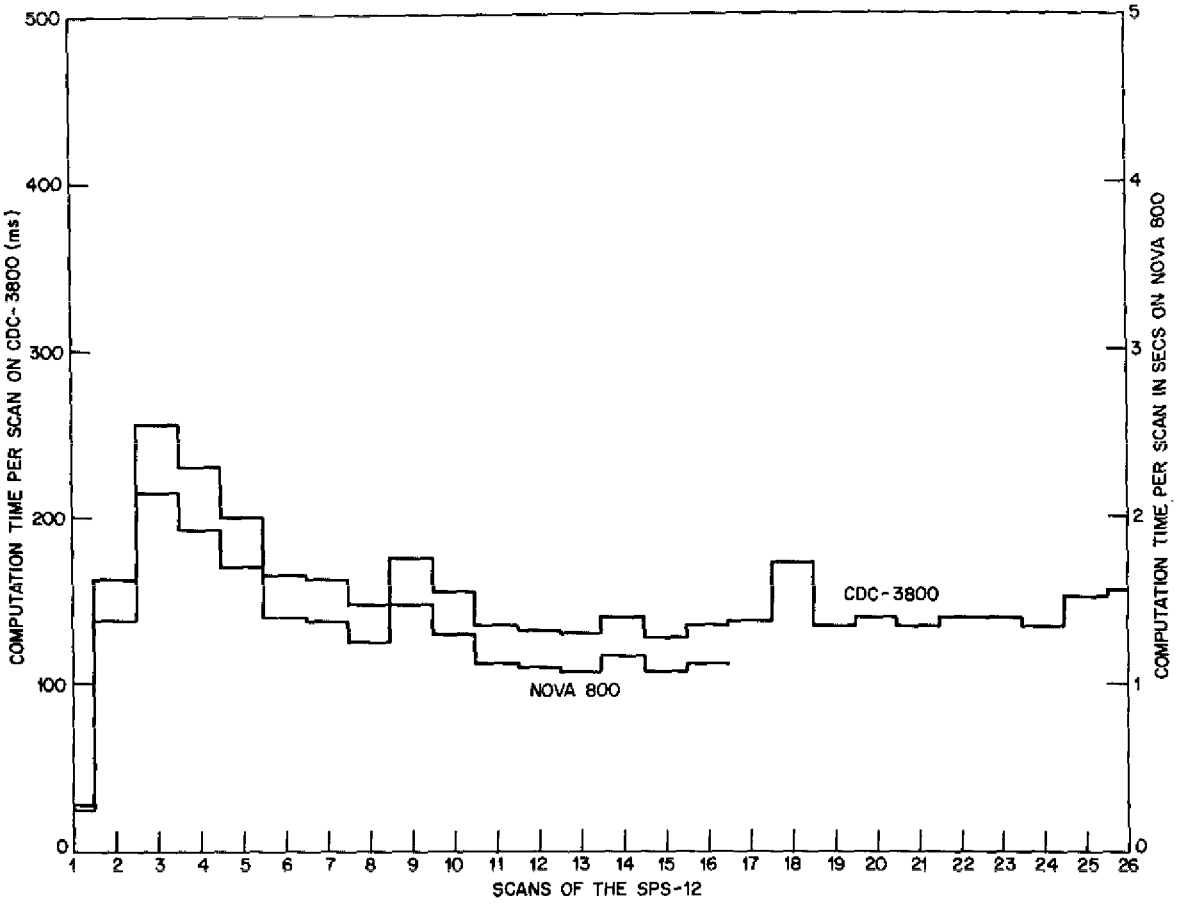


Fig. 10 — Computation time per scan of the SPS-12; low target density

9.5.2. Medium Target Density — In this simulation the following parameters were used:

N_c = 100 clutter points
N_T = 50 targets
P_c = 0.95 probability of detecting clutters
P_T = 0.90 probability of detecting targets
R_{max} = 106 n.mi.

In Fig. 11 the number of tentative tracks, target tracks, and clutters per scan of the SPS-12 are shown. Again, approximately 25% of the clutters are initially classified as targets. However, most of these bogus targets are eliminated on the eighth scan of the SPS-39. Thus steady state is approached on the eleventh scan of the SPS-12.

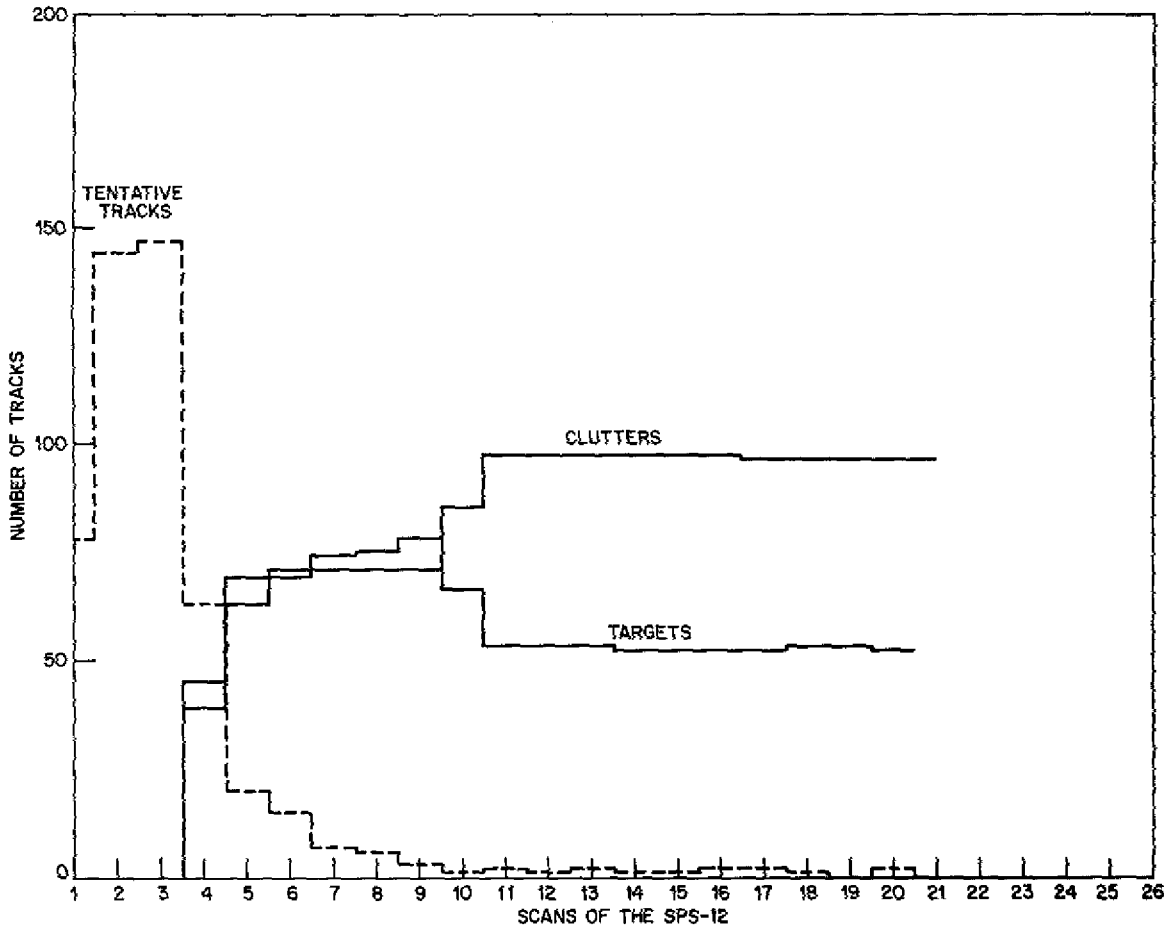


Fig. 11 — Number of tracks (clutter points, targets, and tentative tracks) vs scans of the SPS-12; medium target density

The processing time is shown in Fig. 12. The Nova 800 is 9.0 times slower than the CDC 3800. The largest processing time for the Nova occurs on the fourth scan. The processing time is 4.7 s, which is very close to the scan time of 6 s. If there were about 200 detections per scan, the processing would start lagging behind and sectors could be skipped. However, since the steady-state processing time is below 6 s, steady state would eventually be reached.

9.6. Conclusions

From the simulations on the Nova 800, it appears that the tracking program written in Fortran is fast enough to handle tracking loads that one would expect at CBD. Thus, the time and effort required to program the tracking system in assembly language can be avoided. Of course, the program would run several times faster if written in assembly language.

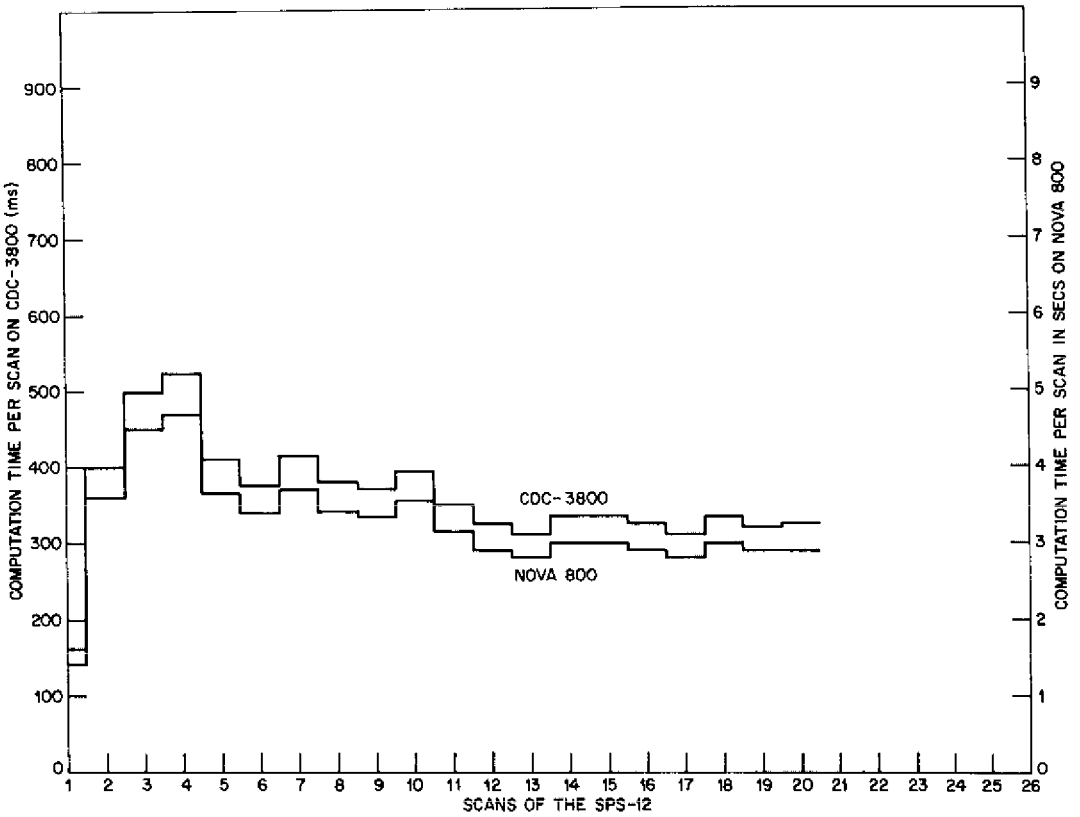


Fig. 12 — Computation time per scan of the SPS-12; medium target density

The tracks that have been declared targets should not be considered "firm" tracks until they have passed the movement test that is performed every eighth scan of the SPS-39. There is not presently anything in the program called a "firm track," but it probably will be added.

10.0. SUMMARY

The tracking system, designed to track targets using detections from two asychronously scanning radars in close proximity, differs from previous single-radar tracking systems in timing, filter update, track initiation, and the use of detections from two radars.

The system is timed with respect to a single clock located outside the computer. Every detection is associated with a time of occurence. In addition, the time of each sector crossing for each radar is sent to the computer. The ideal way of processing the detections is to operate on them sequentially in time. However, the system operates on small azimuth sectors sequentially in time. The detections in the sector located farthest back in time is always operated on first. Thus, detections from one radar in a sector are operated on. This procedure causes very little difficulty.

Since the detections are updating the tracks at nonuniform time intervals, filter coefficients are varied according to the elapsed time between detections. The predicted position is computed using the interval between the time the next radar will be over the target and the current time. This interval is nonuniform from one update to the next.

A track is initiated on detections that do not correlate with anything. After a period of time has passed (which allows the filter to settle) a decision is made on what to do with the new track.

A Fortran program was written for this system and simulated on the Nova 800. It was found that it took about 3 *s* to process 100 point clutters and 50 targets.

REFERENCES

1. G.V. Trunk, B.H. Cantrell, and D.F. Queen, "Basic System Concept for Integrating a 2D and a 3D Radar and Designs of Automatic Detection Systems," NRL Report 7678, Mar. 7, 1974.

2. B.H. Cantrell, G.V. Trunk, and F.D. Queen, "A Detector Design for the SPS-10 Radar," NRL Report 7734, June 19, 1974.

3. K.E. Richeson, "Fleet Systems Report, Radar Detection System Software," SMS-FS-513, MWD-2-351, Nov. 1971, Johns Hopkins University, Applied Physics Laboratory.

4. B.H. Cantrell, "Gain Adjustments of an Alpha-Beta Filter With Random Updates," NRL Report 7647, Dec. 21, 1974.

Appendix A

TRACK AND CLUTTER NUMBER FILES

```
      SUBROUTINE CLTNO(NC, DROPC)
      COMMON/CNO/ LISTC(256), NEXTC, LASTC, FULLC, I12DEL
       INTEGER FULLC, DROPC
      IF(DROPC) 20, 10, 20
   10 LISTC(LASTC)=NC
      LISTC(NC)=0
      LASTC=NC
      FULLC = FULLC+1
      RETURN
   20 NC=NEXTC
      NEXTC=LISTC(NC)
   40 FULLC = FULLC-1
   30 LISTC(NC)=512
      RETURN
      END
```

*Y5XT$$

```
      SUBROUTINE TRKNO(NT,DROPT)
      COMMON/TNO/LISTT(256),NEXTT,LASTT,FULLT
       INTEGER FULLT,DROPT
      IF(DROPT) 20,10,20
10    LISTT(LASTT)=NT
      LISTT(NT)=0
      LASTT=NT
      FULLT = FULLT+1
      RETURN
20    NT=NEXTT
      NEXTT=LISTT(NT)
40    FULLT = FULLT-1
30    LISTT(NT) = 512
      RETURN
      END
```

Appendix B

AZIMUTH SECTOR FILES

```
      SUBROUTINE TNEW(NT,I)
      COMMON/TFILE/TBX(64),IDT(256)
       INTEGER TBX
      NL = TBX(I)
      TBX(I) = NT
      IDT(NT) = NL
      RETURN
      END TNEW
```

```
      SUBROUTINE TDROP(NT,I)
      COMMON/TFILE/TBX(64),IDT(256)
       INTEGER TBX
      NL=TBX(I)
      IF(NL-NT) 20,10,20
10    TBX(I)=IDT(NT)
      IDT(NT)=0
      NT = TBX(I)
      RETURN
20    NE=IDT(NL)
      IF(NT-NE) 30,40,30
30    NL=NE
      GO TO 20
40    IDT(NL)=IDT(NE)
      NT = IDT(NT)
      IDT(NE)=0
      RETURN
      END
```

Appendix C

SMOOTHING FILTER

```
      SUBROUTINE FILTR(RM, AM, DELT, NT)
      COMMON/TPAR2/RS(256), AS(256), VAS(256)
      COMMON/TPAR1/RPT(256), APT(256), ES(256), VRS(256), OUT(256), ISTA(12)

      COMMON/TPAR7/RALPA(128), AALPA(128), RBETA(128), ABETA(128)
       INTEGER RALPA, AALPA, RBETA, ABETA, DELT, DELTQ, RM, AM
      INTEGER RPT, APT, RS, AS, VRS, VAS, ES, OUT
C-----TIME GAP FUNCTION OF ALPHA-BETA TABLES.
      DELTQ = DELT/20 + 1
      IF(DELTQ-64) 10, 10, 2
    2 DELTQ = (DELT+1260)/40 + 1
      IF(DELTQ-96) 10, 10, 4
    4 DELTQ = (DELT+5060)/80 + 1
      IF(DELTQ-128) 10, 10, 6
    6 DELTQ = 128
C     4096 FT/SEC=2**15
C     1 COUNT = 31.25 FEET.
C     8 DEG/SEC = 2**15
C     360 DEG = 2**15
C     1 COUNT = 8 MS
   10 K=RM-RPT(NT)
      RS(NT)=RPT(NT)+(RALPA(DELTQ)*K)/32
      IT = AM-APT(NT)
      ITS = IABS(IT)
      IF(ITS-16384) 40, 40, 20
   20 ITS = 32767-ITS
      IF(IT.LT.0) GO TO 30
      IT = -ITS
      GO TO 40
   30 IT = ITS
   40 CONTINUE
      M=APT(NT)/2+(AALPA(DELTQ)*IT)/64
      IF (M-16384) 42, 45, 45
   42 IF (M) 43, 50, 50
   43 M=M+16384
      GO TO 50
   45 M=M-16384
   50 AS(NT)=M+M
      IF (DELT-100) 100, 60, 60
   60 IF (DELT-300) 70, 70, 80
   70 VRS(NT)=VRS(NT)+122*((8*RBETA(DELTQ)*K)/DELT)
      VAS(NT)=VAS(NT)+22*((8*ABETA(DELTQ)*IT)/DELT)
      GO TO 100
   80 VRS(NT)=VRS(NT)+244*((RBETA(DELTQ)*K)/(DELT/4))
      VAS(NT)=VAS(NT)+44*((ABETA(DELTQ)*IT)/(DELT/4))
  100 CONTINUE
      RETURN
      END
```

Appendix D

TIME TO NEXT UPDATE

```
      SUBROUTINE TME12(TH,T,IG,NT)
      COMMON/VEL/ V12,V39
      COMMON/TPAR2/RS(256),AS(256),VAS(256)
      COMMON/TPAR3/TT12(256),TT39(256),TTL12(256),TTL39(256),TT(256)
       INTEGER V12,V39,T,TH,RS,AS,VAS
       INTEGER TT12,TT39,TTL12,TTL39,TT
      KT12=32767/(((V12-VAS(NT))/22)/250)
      M=TH/2+KT12/2
      IF (M-16384) 20,10,10
10    M=M-16384
20    TTL12(NT)=M+M
      T = TTL39(NT) - TH
      IF(IABS(T).GT.16384) T = 32767+T
      IF(T.LT.0) T = 0
      IF(T.GT.KT12) T = KT12
      RETURN
      END
```

```
      SUBROUTINE TME39(TH,T,IG,NT)
      COMMON/VEL/ V12,V39
      COMMON/TPAR2/RS(256),AS(256),VAS(256)
      COMMON/TPAR3/TT12(256),TT39(256),TTL12(256),TTL39(256),TT(256)
       INTEGER V12,V39,T,TH,RS,AS,VAS
       INTEGER TT12,TT39,TTL12,TTL39,TT
      KT39=32767/(((V39-VAS(NT))/22)/250)
      M=TH/2+KT39/2
      IF (M-16384) 20,10,10
10    M=M-16384
20    TTL39(NT)=M+M
      T = TTL12(NT) - TH
      IF(IABS(T).GT.16384) T = 32767+T
      IF(T.LT.0) T = 0
      IF(T.GT.KT39) T = KT39
      RETURN
      END
```

Appendix E

EXECUTIVE

```
      SUBROUTINE EXCUT
      COMMON/ALP/IPAR1,IPAR2,IOPER,NUM,JTAR(32),NHAND,ISTART,NCATCH
      COMMON/ELE/ NDESTAR(4),IAZIM(16),NTARPR(8),NCOUNT
      COMMON/KON/ KONST
      COMMON/DAT12/RM12(256),AM12(256),TM12(256),TAG12(256)
      COMMON/IN12/MRK12(64),NP12(64),NB12(64),P1239(64),P1210(64),I12T
      COMMON/DAT39/ RM39(256),AM39(256),TM39(256),TAG39(256),EM39(256)
      COMMON/IN39/ MRK39(64),NP39(64),NB39(64),P3912(64),P3910(64),I39T

      COMMON/VEL/ V12,V39
      COMMON/SECT/ I12D,I39D
      COMMON/CLT/RPC(256),APC(256),TC12(256),TC39(256)
      COMMON/CPAR1/ TCMAX,TCLAG,CRC,CAC
      COMMON/CFILE/CBX(64),IDC(256)
      COMMON/CNO/ LISTC(256),NEXTC,LASTC,FULLC,I12DEL
      COMMON/TFILE/TBX(64),IDT(256)
      COMMON/TNO/LISTT(256),NEXTT,LASTT,FULLT
      COMMON/TPAR1/RPT(256),APT(256),ES(256),VRS(256),OUT(256),ISTA(12)

      COMMON/TPAR2/RS(256),AS(256),VAS(256)
      COMMON/TPAR3/TT12(256),TT39(256),TTL12(256),TTL39(256),TT(256)
      COMMON/TPAR4/KT(256),TF(256),NTARGET,NELEV,ISKIP
      COMMON/TPAR5/CRT(8,2),CAT(8,2,16),TTMAX,TTLAG
      COMMON/TPAR6/VRMIN,VAMIN,TNMAX,TFIX
      COMMON/TPAR7/RALPA(128),AALPA(128),RBETA(128),ABETA(128)
      COMMON/TPAR8/NR(256),NMOD
      COMMON/BUF/IBUF(30),III(1000),NST
       INTEGER CBX
       INTEGER RM12,AM12,TM12,TAG12
       INTEGER RM39,AM39,TM39,TAG39,EM39
       INTEGER V12,V39
       INTEGER P1239,P1210
       INTEGER P3912,P3910
       INTEGER RPC,APC,TC12,TC39
       INTEGER TCMAX,TCLAG,CRC,CAC
       INTEGER OUT,CRT,CAT,TTMAX,TTLAG
       INTEGER FULLC,DROPC
       INTEGER FULLT,DROPT
       INTEGER RALPA,AALPA,RBETA,ABETA
       INTEGER TF
       INTEGER TBX
       INTEGER DELT,T,VRMIN,VAMIN,TNMAX,TFIX
       INTEGER TT12,TT39,TTL12,TTL39,TT
       INTEGER RPT,APT,RS,AS,VRS,VAS,ES
      ACCEPT "REAL(0) OR SIMULATED(1) DATA.      ",IREAL
      IF(IREAL) 300,300,301
  300 II = I12T-I39T
      IF(II) 310,320,320
  310 II = 64+II
  320 IF(II-4) 300,330,330
  330 IF(II-40) 340,300,300
  340 I12D = I12T
      I39D = I39T
      GO TO 1
  301 ACCEPT "INPUT FIRST SCAN PRINTED OUT.      ",JSCAN
      ACCEPT "INPUT NUMBER OF SECTORS READ.      ",NSEC
      NST=1000
```

```
      CALL GTDA
      I12T=I12D
      I39T=I39D
      I12C = 9
      I39C = 9
    1 ISCAN=0
      IELEV = 0
C
C     HOW MANY SECTORS ARE WE LAGGING BEHIND
C
    5 I12DEL=I12T-I12D
      ICK = IABS(I39T-61)
      IF(ICK-1) 6,6,7
    6 IF(IELEV) 7,666,7
  666 CALL ELEV
      IELEV = 1
    7 IF(I12DEL) 8,9,9
    8 I12DEL = I12DEL + 64
    9 IF(I12DEL-20) 10,10,4
    4 IF(IREAL) 45,45,44
   44 WRITE (10,50)
   50 FORMAT (1H0," LAGGING BEHIND MORE THAN 12 SECTORS")
   45 CONTINUE
      IJ = I39T - 15
      IF(IJ) 497,497,498
  497 IJ = IJ+64
  498 NDEL = MRK39(I39T+1)-MRK39(IJ)
      IF(NDEL) 499,501,501
  499 NDEL = NDEL+32767
  501 V39 = 32000/(NDEL/16) + 3*(V39/4)
      I39C = 8
      IJ = I12T-15
      IF(IJ) 597,597,598
  597 IJ = IJ+64
  598 NDEL = MRK12(I12T+1)-MRK12(IJ)
      IF(NDEL) 599,601,601
  599 NDEL = NDEL+32767
  601 V12 = 32000/(NDEL/16) + 3*(V12/4)
      I12C = 8
      IS12=32767/(V12/250)
      IS39=32767/(V39/250)
      IEND=I12D+I12DEL-4
      DO 440 I=I12D,IEND
      K=I+1
      IF (K-64) 402,402,401
  401 K=K-64
  402 NT=TBX(K)
  410 IF (NT) 440,440,420
  420 TTL12(NT)=TTL12(NT)+IS12
      NT=IDT(NT)
      GO TO 410
  440 CONTINUE
      IF (K-64) 450,445,445
  445 K=0
  450 I12D=K
```

```
      I39DEL=I39T-I39D
      IF (I39DEL) 451,452,452
  451 I39DEL=I39DEL+64
  452 IEND=I39D+I39DEL-4
      DO 490 I=I39D,IEND
      K=I+1
      IF (K-64) 462,462,461
  461 K=K-64
  462 NT=TBX(K)
  470 IF (NT) 490,490,480
  480 TTL39(NT)=TTL39(NT)+IS39
      NT=IDT(NT)
      GO TO 470
  490 CONTINUE
      IF (K-64) 496,495,495
  495 K=0
  496 I39D=K
   10 IF (I12DEL-5) 2,30,30
    2 CALL ALPNM
      NCATCH=NCATCH+1
      IF (IREAL) 5,5,46
C
C     HAVE PROCESSED ALL DATA
C     READ IN NSEC ADDITIONAL SECTORS
C
   46 IF(NST.LT.800) GO TO 11
      CALL GTDA
   11 CONTINUE
      DO 29 I=1,NSEC
      CALL SHIFT(5)
   12 IF (IBUF(1)-20) 13,20,20
C
C     DATA FROM SPS-12
C
   13 II=IBUF(2)
      I12T=II-1
      MRK12(II)=IBUF(3)
      P1239(II) = IBUF(4)
      ITAR=IBUF(5)
      NB12(II)=ITAR
      JJ = II-1
      IF(JJ.EQ.0) JJ = 16
      N=NP12(JJ)+ITAR
      IF (N-256) 112,111,111
  111 N=N-256
  112 NP12(II)=N
      IF (ITAR.EQ.0) GO TO 29
      N=3*ITAR
      CALL SHIFT(N)
   15 NS=NP12(JJ) + 1
      DO 16 J=1,ITAR
      NS=NS+1
      IF(NS.EQ.257) NS = 1
      RM12(NS)=IBUF(J)
      AM12(NS)=IBUF(J+ITAR)
```

```
      TM12(NS)=IBUF(J+2*ITAR)
   16 CONTINUE
      GO TO 29
C
C     DATA FROM SPS-39
C
   20 II=IBUF(2)
      I39T=II+1
      MRK39(II)=IBUF(3)
      P3912(II) = IBUF(4)
      ITAR=IBUF(5)
      NB39(II)=ITAR
      JJ = II-1
      IF(JJ.EQ.0) JJ = 16
      N=NP39(JJ)+ITAR
      IF (N-256) 212,211,211
  211 N=N-256
  212 NP39(II)=N
      IF (ITAR.EQ.0) GO TO 29
      N=3*ITAR
      CALL SHIFT(N)
   25 NS=NP39(JJ) + 1
      DO 26 J=1,ITAR
      NS=NS+1
      IF(NS.EQ.257) NS = 1
      RM39(NS)=IBUF(J)
      AM39(NS)=IBUF(J+ITAR)
      TM39(NS)=IBUF(J+2*ITAR)
   26 CONTINUE
   29 CONTINUE
      GO TO 5
C
C     DECISION AS TO WHAT RADAR SECTOR TO UPDATE
C
   30 IDIF = MRK12(I12D+1) - MRK39(I39D+1)
      IDIFA=IABS(IDIF)
      IF (IDIFA-16384) 31,31,40
   31 IF (IDIF) 60,60,80
   40 IF (IDIF) 80,60,60
C
C     UPDATE SPS-12
C
   60 IF (I12D) 65,61,65
   61 IF (IQPER) 65,62,65
   62 CALL ALPNM
   65 I12C = I12C-1
      IF(I12C) 73,66,73
   66 IJ = I12T-7
      IF(IJ) 67,67,68
   67 IJ = IJ+64
   68 NDEL = MRK12(I12T+1)-MRK12(IJ)
      IF(NDEL) 69,72,72
   69 NDEL = 32767+NDEL
   72 V12 = 16000/(NDEL/8)+7*(V12/8)
      I12C = 8
```

```
   73 CONTINUE
      CALL MAP12
      CALL TRK12
      CALL NEW12
      I12D=I12D+1
      IF (I12D-64) 5,70,70
   70 I12D=0
      ISCAN = ISCAN+1
      IPRT = MOD(ISCAN,JSCAN)
      IF (IREAL) 5,5,71
   71 IF(IPRT.NE.0) GO TO 5
      WRITE (10,56)
   56 FORMAT(1H1)
      WRITE (10,51) V12,V39,NEXTC,LASTC,ISCAN
   51 FORMAT(" V12 = ",I6," V39 = ",I6,/,
     1      " NEXTC = ",I4," LASTC = ",I4," SCAN = ",I3)
      WRITE (10,57)
   57 FORMAT(1X,/," SCAN      RPT       APT       RS        AS"
     1      ,"      VRS       VAS      TT12",/,"      TT39     TTL12"
     2      ,"     TTL39        TT        KT        TF        NR")
      DO 555 I=1,256
      IF (KT(I ).GT.10) GO TO 555
   55 WRITE (10,150) I,RPT(I),APT(I),RS(I),AS(I),VRS(I),VAS(I),TT12(I),
     1      TT39(I),TTL12(I),TTL39(I),TT(I),KT(I),TF(I),NR(I)
  555 CONTINUE
  150 FORMAT(I4,7I9,/,7I9,/)
      GO TO 5
C
C     UPDATE SPS-39
C
   80 IF (I39D) 85,81,85
   81 IEVEV=0
      NMOD=NMOD+1
      IF (NMOD-8) 85,82,85
   82 NMOD=0
   85 I39C = I39C-1
      IF(I39C) 93,86,93
   86 IJ = I39T-7
      IF(IJ) 87,87,88
   87 IJ = IJ+64
   88 NDEL = MRK39(I39T+1)-MRK39(IJ)
      IF(NDEL) 89,92,92
   89 NDEL = 32767+NDEL
   92 V39 = 16000/(NDEL/8)+7*(V39/8)
      I39C = 8
   93 CONTINUE
      CALL MAP39
      CALL TRK39
      CALL NEW39
      I39D=I39D+1
      IF (I39D-64) 5,90,90
   90 I39D=0
      GO TO 5
      END
```

```
      SUBROUTINE MAP12
      COMMON/KON/ KONST
      COMMON/DAT12/RM12(256),AM12(256),TM12(256),TAG12(256)
      COMMON/IN12/MRK12(64),NP12(64),NB12(64),P1239(64),P1210(64),I12T
      COMMON/VEL/ V12,V39
      COMMON/CPAR1/ TCMAX,TCLAG,CRC,CAC
      COMMON/CLT/RPC(256),APC(256),TC12(256),TC39(256)
      COMMON/CNO/ LISTC(256),NEXTC,LASTC,FULLC,I12DEL
      COMMON/CFILE/CBX(64),IDC(256)
      COMMON/SECT/ I12D,I39D
       INTEGER FULLC,DROPC
       INTEGER CBX
       INTEGER RM12,AM12,TM12,TAG12
       INTEGER V12,V39
       INTEGER P1239,P1210
       INTEGER RPC,APC,TC12,TC39
       INTEGER TCMAX,TCLAG,CRC,CAC
       INTEGER D,DI
       INTEGER DELR,DELA,TH,RM,AM
      IH = I12D + 2
      IF(IH-64) 5,5,2
    2 IH = IH-64
    5 NC=CBX(IH)
   15 CONTINUE
      IF(NC) 10,1000,10
   10 NDEL1 = MRK12(IH) - TC12(NC)
      IF(NDEL1) 20,30,30
   20 NDEL1=32767+NDEL1
   30 IF(NDEL1-150) 900,900,40
   40 IF(NDEL1-TCMAX) 200,200,50
   50 NDEL2 = MRK12(IH) - TC39(NC)
      IF(NDEL2) 60,200,70
   60 NDEL2 = 32767 + NDEL2
   70 IF(NDEL2-TCMAX) 200,200,80
   80 CALL CLTNO(NC,0)
      CALL CDROP(NC,IH)
      GO TO 15
  200 J=0
      D=KONST
  210 IF(J-2) 220,220,400
  220 JK = I12D +J +1
      IF(JK-64) 222,222,221
  221 JK = JK-64
  222 J = J+1
      JB=NP12(JK) + 1
      K=1
  225 IF(K-NB12(JK)) 230,230,210
  230 DELR = IABS(RM12(JB) - RPC(NC))
      IF(DELR-CRC) 240,240,300
  240 DELA = IABS(AM12(JB) - APC(NC))
      IF(DELA-16384) 246,244,244
  244 DELA = 32767-DELA
  246 CONTINUE
      IF(DELA-CAC) 250,250,300
  250 NDI = 64*DELR/CRC
```

```
      ND2 = 64*DELA/CAC
      ND1=ND1*ND1
      ND2=ND2*ND2
      DI=ND1+ND2
      IF(DI-D) 260,260,270
  260 RM=RM12(JB)
      AM=AM12(JB)
      TH=TM12(JB)
      D=DI
  270 JT=NP12(JK)-NB12(JK) + 2
      IF(JT) 280,280,290
  280 JT=JT+256
  290 RM12(JB)=RM12(JT)
      AM12(JB)=AM12(JT)
      TM12(JB)=TM12(JT)
      NB12(JK)=NB12(JK)-1
      GO TO 225
  300 K=K+1
      JB=JB-1
      IF(JB) 310,310,320
  310 JB=JB+256
  320 GO TO 225
  400 IF(D-KONST) 410,900,900
  410 RPC(NC)=RM
      APC(NC)=AM
      TC12(NC)=TH
      ISECT = AM/512+1
      IF(ISECT-IH) 420,900,420
  420 NS = NC
      CALL CDROP(NC,IH)
      CALL CNEW(NS,ISECT)
      GO TO 15
  900 NC=IDC(NC)
      GO TO 15
 1000 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE MAP39
      COMMON/KON/ KONST
      COMMON/DAT39/ RM39(256),AM39(256),TM39(256),TAG39(256),EM39(256)
      COMMON/IN39/MRK39(64),NP39(64),NB39(64),P3912(64),P3910(64),I39T
      COMMON/VEL/ V12,V39
      COMMON/CPAR1/ TCMAX,TCLAG,CRC,CAC
      COMMON/CLT/RPC(256),APC(256),TC12(256),TC39(256)
      COMMON/CNO/ LISTC(256),NEXTC,LASTC,FULLC,I12DEL
      COMMON/CFILE/CBX(64),IDC(256)
      COMMON/SECT/ I12D,I39D
       INTEGER FULLC,DROPC
       INTEGER CBX
       INTEGER RM39,AM39,TM39,TAG39,EM39
       INTEGER V12,V39
       INTEGER P3912,P3910
       INTEGER RPC,APC,TC12,TC39
       INTEGER TCMAX,TCLAG,CRC,CAC
       INTEGER D,DI
       INTEGER DELR,DELA,TH,RM,AM
      IH = I39D+2
      IF(IH-64) 5,5,2
    2 IH = IH-64
    5 NC=CBX(IH)
   15 CONTINUE
      IF(NC) 10,1000,10
   10 NDEL1 = MRK39(IH) - TC39(NC)
      IF(NDEL1) 20,30,30
   20 NDEL1=32767+NDEL1
   30 IF(NDEL1-150) 900,900,40
   40 IF(NDEL1-TCMAX) 200,200,50
   50 NDEL2 = MRK39(IH)-TC12(NC)
      IF(NDEL2) 60,200,70
   60 NDEL2 = 32767 + NDEL2
   70 IF(NDEL2-TCMAX) 200,200,80
   80 CALL CLTNO(NC,0)
      CALL CDROP(NC,IH)
      GO TO 15
  200 J=0
      D=KONST
  210 IF(J-2) 220,220,400
  220 JK = I39D+J+1
      IF(JK-64) 222,222,221
  221 JK = JK-64
  222 J = J+1
      JB=NP39(JK) + 1
      K=1
  225 IF(K-NB39(JK)) 230,230,210
  230 DELR = IABS(RM39(JB) - RPC(NC))
      IF(DELR-CRC) 240,240,300
  240 DELA = IABS(AM39(JB) - APC(NC))
      IF(DELA-16384) 246,244,244
  244 DELA = 32767-DELA
  246 CONTINUE
      IF(DELA-CAC) 250,250,300
  250 NDI = 64*DELR/CRC
```

```
      ND2 = 64*DELA/CAC
      ND1=ND1*ND1
      ND2=ND2*ND2
      DI=ND1+ND2
      IF(DI-D) 260,260,270
  260 RM=RM39(JB)
      AM=AM39(JB)
      TH=TM39(JB)
      D=DI
  270 JT=NP39(JK)-NB39(JK) + 2
      IF(JT) 280,280,290
  280 JT=JT+256
  290 RM39(JB)=RM39(JT)
      AM39(JB)=AM39(JT)
      TM39(JB)=TM39(JT)
      NB39(JK)=NB39(JK)-1
      EM39(JB) = EM39(JT)
      GO TO 225
  300 K=K+1
      JB=JB-1
      IF(JB) 310,310,320
  310 JB=JB+256
  320 GO TO 225
  400 IF(D-KONST) 410,900,900
  410 RPC(NC)=RM
      APC(NC)=AM
      TC39(NC)=TH
      ISECT = AM/512+1
      IF(ISECT-IH) 420,900,420
  420 NS = NC
      CALL CDROP(NC,IH)
      CALL CNEW(NS,ISECT)
      GO TO 15
  900 NC=IDC(NC)
      GO TO 15
 1000 CONTINUE
      RETURN
      END
```

Appendix G

TRACK UPDATE

```
      SUBROUTINE TRK12
      COMMON/TEST/ISECT,NT,M,IH
      COMMON/ELE/ NDESTAR(4),IAZIM(16),NTARPR(8),NCOUNT
      COMMON/ALP/IPAR1,IPAR2,IOPER,NUM,JTAR(32),NHAND,ISTART,NCATCH
      COMMON/KON/ KONST
      COMMON/DAT12/RM12(256),AM12(256),TM12(256),TAG12(256)
      COMMON/IN12/MRK12(64),NP12(64),NB12(64),P1239(64),P1210(64),I12T
      COMMON/SECT/ I12D,I39D
      COMMON/CPAR1/ TCMAX,TCLAG,CRC,CAC
      COMMON/CLT/RPC(256),APC(256),TC12(256),TC39(256)
      COMMON/CFILE/CBX(64),IDC(256)
      COMMON/CNO/ LISTC(256),NEXTC,LASTC,FULLC,I12DEL
      COMMON/VEL/ V12,V39
      COMMON/TFILE/TBX(64),IDT(256)
      COMMON/TNO/LISTT(256),NEXTT,LASTT,FULLT
      COMMON/TPAR1/RPT(256),APT(256),ES(256),VRS(256),OUT(256),ISTA(12)

      COMMON/TPAR2/RS(256),AS(256),VAS(256)
      COMMON/TPAR3/TT12(256),TT39(256),TTL12(256),TTL39(256),TT(256)
      COMMON/TPAR4/KT(256),TF(256),NTARGET,NELEV,ISKIP
      COMMON/TPAR5/CRT(8,2),CAT(8,2,16),TTMAX,TTLAG
      COMMON/TPAR6/VRMIN,VAMIN,TNMAX,TFIX
       INTEGER RM12,AM12,TM12,TAG12
       INTEGER V12,V39
       INTEGER P1239,P1210
       INTEGER RPT,APT,RS,AS,VRS,VAS,ES
       INTEGER OUT,CRT,CAT,TTMAX,TTLAG
       INTEGER D,DI,RQ,DELR,DELA,TH,RM,AM
       INTEGER TT12,TT39,TTL12,TTL39,TT
       INTEGER DELT,T,VRMIN,VAMIN,TNMAX,TFIX
       INTEGER FULLT,DROPT
       INTEGER TBX
       INTEGER TF
       INTEGER TCMAX,TCLAG,CRC,CAC
       INTEGER FULLC,DROPC
       INTEGER RPC,APC,TC12,TC39
       INTEGER CBX
      IH=I12D+1
      IFLIP=0
    5 NT=TBX(IH)
   15 CONTINUE
      IF(NT) 10,1050,10
   10 IF(KT(NT)-IFLIP) 1000,20,1000
   20 NDEL1=IABS(MRK12(IH)-TTL12(NT))
      IF(NDEL1-16384) 40,30,30
   30 NDEL1=32767-NDEL1
   40 IF(NDEL1-150) 50,50,1070
   50 NDEL2=IABS(MRK12(IH)-TT(NT))
      IF(NDEL2-16384) 70,60,60
   60 NDEL2=32767-NDEL2
   70 NDEL2=NDEL2/625+1
      IF(NDEL2-8) 90,80,80
   80 NDEL2=8
   90 RQ=RPT(NT)/2048+1
      JCRT=CRT(NDEL2,IFLIP+1)
      JCAT=CAT(NDEL2,IFLIP+1,RQ)
```

```
      J=0
      D=KONST
  210 IF(J-3) 220,220,400
  220 JK=I12D+J
      IF (JK-64) 222,222,221
  221 JK=JK-64
      GO TO 224
  222 IF (JK) 223,223,224
  223 JK=64
  224 J=J+1
      JB=NP12(JK) + 1
      K=1
  225 IF(K-NB12(JK)) 230,230,210
  230 DELR=IABS(RM12(JB)-RPT(NT))
      IF(DELR-JCRT) 240,300,300
  240 DELA=IABS(AM12(JB)-APT(NT))
      IF(DELA-16384) 260,250,250
  250 DELA = 32767-DELA
  260 CONTINUE
      IF(DELA-JCAT) 270,300,300
  270 ND1=(63*DELR)/JCRT
      ND2=(63*DELA)/JCAT
      ND1=ND1*ND1
      ND2=ND2*ND2
      DI=ND1+ND2
      TAG12(JB)=1
      IF(DI-D) 280,280,300
  280 JCALL=JB
      JSECT=JK
      D = DI
  300 K=K+1
      JB=JB-1
      IF(JB) 310,310,225
  310 JB=JB+256
      GO TO 225
  400 IF(D-KONST) 410,800,800
  410 RM=RM12(JCALL)
      AM=AM12(JCALL)
      TH=TM12(JCALL)
      JT=NP12(JSECT)-NB12(JSECT)+2
      IF(JT) 420,420,430
  420 JT=256+JT
  430 RM12(JCALL)=RM12(JT)
      AM12(JCALL)=AM12(JT)
      TM12(JCALL)=TM12(JT)
      TAG12(JCALL) = TAG12(JT)
      TAG12(JT) = 0
      NB12(JSECT)=NB12(JSECT)-1
      DELT=IABS(TH-TT(NT))
      IF(DELT-16384) 520,510,510
  510 DELT=32767-DELT
  520 CONTINUE
      CALL FILTR(RM,AM,DELT,NT)
      TT(NT)=TH
      TT12(NT)=TH
```

```
      TH = MRK12(IH)
      CALL TME12(TH,T,IH,NT)
      KK=((T/10)*(VRS(NT)/125))/25
      RPT(NT) = RS(NT) + KK
      KK=(T/32)*(VAS(NT)/187)
      M = AS(NT)/4 + KK/4
      IF (M-8192) 521,523,523
  521 IF (M) 522,524,524
  522 M=M+8192
      GO TO 524
  523 M=M-8192
  524 APT(NT)=4*M
      ISECT=M/128+1
      IF (IFLIP) 600,525,600
  525 IF (NT-NHAND) 530,526,530
  526 CALL ALPNM
  530 IF(ISECT-IH) 540,1000,540
  540 NS = NT
      CALL TDROP(NT,IH)
      CALL TNEW(NS,ISECT)
      GO TO 15
  600 DELT=IABS(TH        -TF(NT))
      IF(DELT-16384) 620,610,610
  610 DELT=32767-DELT
  620 IF(DELT-TFIX) 530,530,630
  630 IF(IABS(VRS(NT)) - VRMIN) 640,670,670
  640 IF(IABS(VAS(NT)) - VAMIN) 690,670,670
  670 KT(NT)=0
      TF(NT)=0
      NTARGET=NTARGET+1
      GO TO 530
  690 IF(FULLC) 720,720,700
  700 CALL CLTNO(NC,1)
      RPC(NC)=RS(NT)
      APC(NC)=AS(NT)
      TC12(NC)=TH
      TC39(NC)=TT39(NT)
      CALL CNEW(NC,ISECT)
  720 CALL TRKNO(NT,0)
      OUT(NT)=-1
      KT(NT)=63
      CALL TDROP(NT,IH)
      GO TO 15
C
C-----NO CORELLATION WITH THIS TRACK.
C
  800 CONTINUE
      TH = MRK12(IH)
      DELT=TH-TT(NT)
      IF (DELT) 870,880,880
  870 DELT=32767+DELT
  880 CONTINUE
      CALL TME12(TH,T,IH,NT)
      T=T+DELT
      IF (IFLIP) 900,890,900
```

```
890   IF (T-TTMAX) 891,891,889
900   IF (T-TNMAX) 910,910,720
889   NTARGET=NTARGET-1
      GO TO 720
891   KTARG = TH-TT12(NT)
      IF(KTARG) 881,882,882
 881  KTARG = 32767 + KTARG
 882  IF(KTARG-TTMAX) 885,885,883
 883  TT12(NT) = TH-TTLAG
      IF(TT12(NT)) 884,885,885
 884  TT12(NT) = 32767 + TT12(NT)
 885  CONTINUE
910   KK=((T/40)*(VRS(NT)/148))/6
      RPT(NT) = RS(NT) + KK
      KK=(T/50)*(VAS(NT)/112)
      M = AS(NT)/4 + KK/4
      IF (M-8192) 921,923,923
921   IF (M) 922,924,924
922   M=M+8192
      GO TO 924
923   M=M-8192
924   ISECT=M/128+1
      APT(NT)=4*M
      IF(ISECT-IH) 540,1000,540
1000  NT=IDT(NT)
      GO TO 15
1050  IF(IFLIP) 1100,1060,1100
1060  IFLIP = 1
      GO TO 5
1070  CONTINUE
      M=MRK12(IH)/2+32767/(V12/125)
      IF (M-16384) 1075,1071,1071
1071  M=M-16384
1075  TTL12(NT)=M+M
      NT = IDT(NT)
      GO TO 15
1100  CONTINUE
      RETURN
      END
```

```
      SUBROUTINE TRK39
      COMMON/ALP/IPAR1,IPAR2,IOPER,NUM,JTAR(32),NHAND,ISTART,NCATCH
      COMMON/ELE/ NDESTAR(4),IAZIM(16),NTARPR(8),NCOUNT
      COMMON/KON/ KONST
      COMMON/DAT39/ RM39(256),AM39(256),TM39(256),TAG39(256),EM39(256)
      COMMON/SECT/ I12D,I39D
      COMMON/CPAR1/ TCMAX,TCLAG,CRC,CAC
      COMMON/IN39/MRK39(64),NP39(64),NB39(64),P3912(64),P3910(64),I39T
      COMMON/CLT/RPC(256),APC(256),TC12(256),TC39(256)
      COMMON/CFILE/CBX(64),IDC(256)
      COMMON/CNO/ LISTC(256),NEXTC,LASTC,FULLC,I12DEL
      COMMON/VEL/ V12,V39
      COMMON/TFILE/TBX(64),IDT(256)
      COMMON/TNO/LISTT(256),NEXTT,LASTT,FULLT
      COMMON/TPAR1/RPT(256),APT(256),ES(256),VRS(256),OUT(256),ISTA(12)

      COMMON/TPAR2/RS(256),AS(256),VAS(256)
      COMMON/TPAR3/TT12(256),TT39(256),TTL12(256),TTL39(256),TT(256)
      COMMON/TPAR4/KT(256),TF(256),NTARGET,NELEV,ISKIP
      COMMON/TPAR5/CRT(8,2),CAT(8,2,16),TTMAX,TTLAG
      COMMON/TPAR6/VRMIN,VAMIN,TNMAX,TFIX
      COMMON/TPAR8/NR(256),NMOD
       INTEGER RM39,AM39,TM39,TAG39,EM39
       INTEGER V12,V39
       INTEGER P3912,P3910
       INTEGER RPT,APT,RS,AS,VRS,VAS,ES
       INTEGER OUT,CRT,CAT,TTMAX,TTLAG
       INTEGER D,DI,RQ,DELR,DELA,TH,RM,AM
       INTEGER TT12,TT39,TTL12,TTL39,TT
       INTEGER DELT,T,VRMIN,VAMIN,TNMAX,TFIX
       INTEGER FULLT,DROPT
       INTEGER TBX
       INTEGER TF
       INTEGER TCMAX,TCLAG,CRC,CAC
       INTEGER FULLC,DROPC
       INTEGER RPC,APC,TC12,TC39
       INTEGER CBX
      IH=I39D+1
      IFLIP=0
    5 NT=TBX(IH)
   15 CONTINUE
      IF(NT) 10,1050,10
   10 IF(KT(NT)-IFLIP) 1000,20,1000
   20 NDEL1=IABS(MRK39(IH)-TTL39(NT))
      IF(NDEL1-16384) 40,30,30
   30 NDEL1=32767-NDEL1
   40 IF(NDEL1-150) 50,50,1070
   50 NDEL2=IABS(MRK39(IH)-TT(NT))
      IF(NDEL2-16384) 70,60,60
   60 NDEL2=32767-NDEL2
   70 NDEL2=NDEL2/625+1
      IF(NDEL2-8) 90,80,80
   80 NDEL2=8
   90 RQ=RPT(NT)/2048+1
      JCRT=CRT(NDEL2,IFLIP+1)
      JCAT=CAT(NDEL2,IFLIP+1,RQ)
```

```
      J=0
      D=KONST
  210 IF(J-3) 220,220,400
  220 JK=I39D+J
      IF (JK-64) 222,222,221
  221 JK=JK-64
      GO TO 224
  222 IF (JK) 223,223,224
  223 JK=64
  224 J=J+1
      JB=NP39(JK) + 1
      K=1
  225 IF(K-NB39(JK)) 230,230,210
  230 DELR=IABS(RM39(JB)-RPT(NT))
      IF(DELR-JCRT) 240,300,300
  240 DELA=IABS(AM39(JB)-APT(NT))
      IF(DELA-16384) 260,250,250
  250 DELA = 32767-DELA
  260 CONTINUE
      IF(DELA-JCAT) 270,300,300
  270 ND1=(63*DELR)/JCRT
      ND2=(63*DELA)/JCAT
      ND1=ND1*ND1
      ND2=ND2*ND2
      DI=ND1+ND2
      TAG39(JB)=1
      IF(DI-D) 280,280,300
  280 JCALL=JB
      JSECT=JK
      D = DI
  300 K=K+1
      JB=JB-1
      IF(JB) 310,310,225
  310 JB=JB+256
      GO TO 225
  400 IF(D-KONST) 410,800,800
  410 RM=RM39(JCALL)
      AM=AM39(JCALL)
      ES(NT)=EM39(JCALL)
      TH=TM39(JCALL)
      JT=NP39(JSECT)-NB39(JSECT)+2
      IF(JT) 420,420,430
  420 JT=256+JT
  430 RM39(JCALL)=RM39(JT)
      AM39(JCALL)=AM39(JT)
      EM39(JCALL)=EM39(JT)
      TM39(JCALL)=TM39(JT)
      TAG39(JCALL) = TAG39(JT)
      TAG39(JT) = 0
      NB39(JSECT)=NB39(JSECT)-1
      DELT=IABS(TH-TT(NT))
      IF(DELT-16384) 520,510,510
  510 DELT=32767-DELT
  520 CONTINUE
      CALL FILTR(RM,AM,DELT,NT)
```

```
      TT(NT)=TH
      TT39(NT)=TH
      TH = MRK39(IH)
      CALL TME39(TH,T,IH,NT)
      KK=((T/10)*(VRS(NT)/125))/25
      RPT(NT) = RS(NT) + KK
      KK=(T/32)*(VAS(NT)/187)
      M = AS(NT)/4 + KK/4
      IF (M-8192) 521,523,523
  521 IF (M) 522,524,524
  522 M=M+8192
      GO TO 524
  523 M=M-8192
  524 APT(NT)=4*M
      ISECT=M/128+1
      IF (IFLIP) 600,525,600
  525 OUT(NT)=24576
C     SET OUT
  526 KTARG=TH-TT12(NT)
      IF (KTARG) 527,528,528
  527 KTARG=KTARG+32767
  528 IF (KTARG-TTMAX) 531,531,529
  529 OUT(NT)=8192
  531 IF (NT-NHAND) 533,532,533
  532 OUT(NT)=OUT(NT)+1024
      CALL ALPNM
  533 IF (TF(NT)) 539,539,534
  534 OUT(NT)=OUT(NT)+512
      IF (TF(NT)-1) 535,535,538
  535 IF (NCOUNT-8) 536,539,539
  536 NCOUNT=NCOUNT+1
      NTARPR(NCOUNT)=NT
      TF(NT)=5
  538 TF(NT) = TF(NT)-1
  539 IF (NMOD) 530,541,530
  541 IF (IABS(RS(NT)-NR(NT)).LT.97) GO TO 685
      NR(NT)=RS(NT)
  530 IF(ISECT-IH) 540,1000,540
  540 NS = NT
      CALL TDROP(NT,IH)
      CALL TNEW(NS,ISECT)
      GO TO 15
  600 DELT=IABS(TH        -TF(NT))
      IF(DELT-16384) 620,610,610
  610 DELT=32767-DELT
  620 IF(DELT-TFIX) 530,530,630
  630 IF(IABS(VRS(NT)) - VRMIN) 640,670,670
  640 IF(IABS(VAS(NT)) - VAMIN) 690,670,670
  670 KT(NT)=0
      TF(NT)=0
      NTARGET=NTARGET+1
      GO TO 530
  685 NTARGET=NTARGET-1
  690 IF(FULLC) 720,720,700
C     TRANSFER OF A TRACK TO THE CLUTTER FILE
```

```
  700 CALL CLTNO(NC,1)
      RPC(NC)=RS(NT)
      APC(NC)=AS(NT)
      TC39(NC)=TH
      TC12(NC)=TT12(NT)
      CALL CNEW(NC,ISECT)
C     DROP A TRACK
  720 CALL TRKNO(NT,0)
      KT(NT)=63
      OUT(NT)=-1
      CALL TDROP(NT,IH)
      GO TO 15
C
C-----NO CORELLATION WITH THIS TRACK.
C
  800 CONTINUE
      TH = MRK39(IH)
      DELT=TH-TT(NT)
      IF (DELT) 870,880,880
  870 DELT=32767+DELT
  880 CONTINUE
      CALL TME39(TH,T,IH,NT)
      T=T+DELT
      IF (IFLIP) 900,890,900
 890  IF (T-TTMAX) 891,891,889
 900  IF (T-TNMAX) 910,910,720
 889  NTARGET=NTARGET-1
      GO TO 720
 891  OUT(NT)=24576
      KTARG = TH-TT39(NT)
      IF(KTARG) 881,882,882
  881 KTARG = 32767 + KTARG
  882 IF(KTARG-TTMAX) 885,885,883
  883 TT39(NT) = TH-TTLAG
      OUT(NT)=16384
      IF(TT39(NT)) 884,885,885
  884 TT39(NT) = 32767 + TT39(NT)
  885 CONTINUE
 910  KK=((T/40)*(VRS(NT)/148))/6
      RPT(NT) = RS(NT) + KK
      KK=(T/50)*(VAS(NT)/112)
      M = AS(NT)/4 + KK/4
      IF (M-8192) 921,923,923
 921  IF (M) 922,924,924
 922  M=M+8192
      GO TO 924
 923  M=M-8192
 924   APT(NT) = 4*M
      ISECT=M/128+1
      IF (IFLIP) 530,526,530
 1000 NT=IDT(NT)
      GO TO 15
 1050 IF(IFLIP) 1100,1060,1100
 1060 IFLIP = 1
      GO TO 5
```

```
1070 CONTINUE
     M=MRK39(IH)/2+32767/(V39/125)
     IF (M-16384) 1075,1071,1071
1071 M=M-16384
1075 TTL39(NT)=M+M
     NT = IDT(NT)
     GO TO 15
1100 CONTINUE
     RETURN
     END
```

Appendix H

TRACK INITIATION

```
      SUBROUTINE NEW12
      COMMON/DAT12/RM12(256),AM12(256),TM12(256),TAG12(256)
      COMMON/IN12/MRK12(64),NP12(64),NB12(64),P1239(64),P1210(64),I12T
      COMMON/VEL/ V12,V39
      COMMON/TFILE/TBX(64),IDT(256)
      COMMON/TNO/LIST(256),NEXTT,LASTT,FULLT
      COMMON/SECT/ I12D,I39D
      COMMON/TPAR1/RPT(256),APT(256),ES(256),VRS(256),OUT(256),ISTA(12)

      COMMON/TPAR2/RS(256),AS(256),VAS(256)
      COMMON/TPAR3/TT12(256),TT39(256),TTL12(256),TTL39(256),TT(256)
      COMMON/TPAR4/KT(256),TF(256),NTARGET,NELEV,ISKIP
      COMMON/TPAR5/CRT(8,2),CAT(8,2,16),TTMAX,TTLAG
      COMMON/TPAR8/NR(256),NMOD
       INTEGER P1239,P1210
       INTEGER OUT,CRT,CAT,TTMAX,TTLAG
       INTEGER TF
       INTEGER RM12,AM12,TM12,TAG12
       INTEGER V12,V39
       INTEGER RPT,APT,RS,AS,VRS,VAS,ES
       INTEGER TT12,TT39,TTL12,TTL39,TT
       INTEGER FULLT,DROPT
       INTEGER TBX
      IH=I12D
      IF(IH) 10,10,20
   10 IH=64
   20 JB=NP12(IH) + 1
      K=1
   50 IF(K-NB12(IH)) 60,60,200
   60 IF(TAG12(JB)) 70,80,70
   70 TAG12(JB)=0
      GO TO 100
   80 IF (FULLT) 100,100,90
   90 CALL TRKNO(NT,1)
      KR=RM12(JB)
      KA=AM12(JB)
      RPT(NT)=KR
      RS(NT)=KR
      NR(NT)=KR
      APT(NT)=KA
      AS(NT)=KA
      ES(NT)=0
      VRS(NT)=0
      VAS(NT)=0
      NTIM=TM12(JB)
      M=NTIM/2+32767/(V12/125)
      IF (M-16384) 94,93,93
   93 M=M-16384
   94 TTL12(NT)=M+M
      TT12(NT)=NTIM
      TT(NT)=NTIM
      TF(NT)=NTIM
      TT39(NT) = NTIM      - TTLAG
      IF(TT39(NT)) 91,92,92
   91 TT39(NT) = 32767+TT39(NT)
   92 CONTINUE
```

```
      II=(KA-P1239(IM))/2
      IF (II) 95,96,96
95    II=II+16384
96    II=II/(V39/250)
      II=II+NTIM/2
      IF (II-16384) 98,97,97
97    II=II-16384
98    TTL39(NT)=II+II
      KT(NT)=1
      ISECT=KA/512+1
      CALL TNEW(NT,ISECT)
100   K=K+1
      JB=JB-1
      IF(JB) 150,150,50
150   JB=JB+256
      GO TO 50
200   CONTINUE
      RETURN
      END
```

```
      SUBROUTINE NEW39
      COMMON/DAT39/ RM39(256),AM39(256),TM39(256),TAG39(256),EM39(256)
      COMMON/IN39/MRK39(64),NP39(64),NB39(64),P3912(64),P3910(64),I39T
      COMMON/VEL/ V12,V39
      COMMON/TFILE/TBX(64),IDT(256)
      COMMON/TNO/LISTT(256),NEXTT,LASTT,FULLT
      COMMON/SECT/ I12D,I39D
      COMMON/TPAR1/RPT(256),APT(256),ES(256),VRS(256),OUT(256),ISTA(12)

      COMMON/TPAR2/RS(256),AS(256),VAS(256)
      COMMON/TPAR3/TT12(256),TT39(256),TTL12(256),TTL39(256),TT(256)
      COMMON/TPAR4/KT(256),TF(256),NTARGET,NELEV,ISKIP
      COMMON/TPAR5/CRT(8,2),CAT(8,2,16),TTMAX,TTLAG
      COMMON/TPAR8/NR(256),NMOD
       INTEGER P3912,P3910
       INTEGER OUT,CRT,CAT,TTMAX,TTLAG
       INTEGER TF
       INTEGER RM39,AM39,TM39,TAG39,EM39
       INTEGER V12,V39
       INTEGER RPT,APT,RS,AS,VRS,VAS,ES
       INTEGER TT12,TT39,TTL12,TTL39,TT
       INTEGER FULLT,DROPT
       INTEGER TBX
      IH=I39D
      IF(IH) 10,10,20
   10 IH=64
   20 JB=NP39(IH)+1
      K=1
   50 IF(K-NB39(IH)) 60,60,200
   60 IF(TAG39(JB)) 70,80,70
   70 TAG39(JB)=0
      GO TO 100
   80 IF(FULLT) 100,100,90
   90 CALL TRKNO(NT,1)
      KR=RM39(JB)
      KA=AM39(JB)
      RPT(NT)=KR
      RS(NT)=KR
      NR(NT)=KR
      APT(NT)=KA
      AS(NT)=KA
      ES(NT)=EM39(JB)
      VRS(NT)=0
      VAS(NT)=0
      NTIM=TM39(JB)
      M=NTIM/2+32767/(V39/125)
      IF (M-16384) 94,93,93
   93 M=M-16384
   94 TTL39(NT)=M+M
      TT39(NT)=NTIM
      TT(NT)=NTIM
      TF(NT)=NTIM
      TT12(NT) = NTIM      - TTLAG
      IF(TT12(NT)) 91,92,92
   91 TT12(NT) = 32767+TT12(NT)
   92 CONTINUE
```

```
      II=(KA-P3912(IH))/2
      IF (II) 95,96,96
95    II=II+16384
96    II=II/(V12/250)
      II=II+NTIM/2
      IF (II-16384) 98,97,97
97    II=II-16384
98    TTL12(NT)=II+II
      KT(NT)=1
      ISECT=KA/512+1
      CALL TNEW(NT,ISECT)
100   K=K+1
      JB=JB-1
      IF(JB) 150,150,50
150   JB=JB+256
      GO TO 50
200   CONTINUE
      RETURN
      END
```

Appendix I

ALPHANUMERIC DISPLAY

```
      SUBROUTINE ALPNM
      COMMON/ALP/IPAR1,IPAR2,IOPER,NUM,JTAR(32),NHAND,ISTART,NCATCH
      COMMON/TPAR1/RPT(256),APT(256),ES(256),VRS(256),OUT(256),ISTA(12)
     1
      COMMON/TPAR2/RS(256),AS(256),VAS(256)
      COMMON/TPAR3/TT12(256),TT39(256),TTL12(256),TTL39(256),TT(256)
      COMMON/TPAR4/KT(256),TF(256),NTARGET,NELEV,ISKIP
      COMMON/TFILE/TBX(64),IDT(256)
      INTEGER TT12,TT39,TTL12,TTL39,TT
      INTEGER RPT,APT,ES,VRS,OUT,RS,AS,VAS,TF,TBX
      IF (IOPER) 1000,1,1
    1 NUM=0
      IOPER=IOPER+1
      GO TO (100,200,900,106,900,600,900,900,950),IOPER
C     TARGET HANDOFF: IOPER=1,IPAR1=TARGET,IPAR2=1(KILL REQUEST)
  100 IF(IPAR2-1) 105,101,105
  101 NHAND=0
      GO TO 950
  105 NHAND=IPAR1+1
C     TARGET PARAMETERS: IOPER=4,IPAR1=TARGET
  106 I=IPAR1+1
      NUM=6
      JTAR(1)=RS(I)
      JTAR(2)=AS(I)
      JTAR(3)=ES(I)
      JTAR(4)=TT(I)
      JTAR(5)=VRS(I)
      JTAR(6)=VAS(I)
      GO TO 950
C     TARGETS IN AZIMUTH SECTOR:IOPER=2,IPAR1=A1,IPAR2=A2
  200 ISEC1=IPAR1/512+1
      IF (ISTART-1) 203,205,203
  203 ISEC1=ISTART
  205 ISEC2=IPAR2/512+1
      IF (ISEC2-ISEC1) 206,208,208
  206 ISEC2=ISEC2+64
  208 CONTINUE
      DO 250 I=ISEC1,ISEC2
      K=I
      IF (K-64) 211,211,209
  209 K=K-64
  211 NT=TBX(K)
  210 IF (NT) 250,250,215
  215 IF (KT(NT)) 240,220,240
  220 NUM=NUM+1
      JTAR(NUM)=NT-1
      IF (NUM-32) 240,945,945
  240 NT=IDT(NT)
      GO TO 210
  250 CONTINUE
      ISTART=1
      GO TO 950
C      TENTATIVE TRACKS: IOPER=6
  600 DO 650 I=ISTART,64
      NT=TBX(I)
  610 IF (NT) 650,650,615
```

```
  615  IF (KT(NT)-1) 640,620,640
  620  NUM=NUM+1
       JTAR(NUM)=NT-1
       IF(NUM-32) 640,945,945
  640  NT=IDT(NT)
       GO TO 610
  650  CONTINUE
       ISTART=1
       GO TO 950
C      :IOPER=3,TARGETS INSIDE(IPAR2=1) OR OUTSIDE(IPAR2=2) R(IPAR1)
C      LIST TOTAL TRACK FILE: IOPER=5
C      LIST HIGH CLOSING VELOCITY TARGETS: IOPER=7,IPAR1=VELOCITY
C      LIST TARGETS UNDER ELEVATION SEARCH: IOPER=8
  900  DO 940 I=ISTART,64
       NT=TBX(I)
  910  IF (NT) 940,940,915
  915  IF (KT(NT)) 930,920,930
  920  GO TO (950,950,300,950,925,950,700,800,950),IOPER
  300  IF (RPT(NT)-IPAR1) 325,325,330
  325  IF (IPAR2-1) 930,925,930
  330  IF (IPAR2-2) 930,925,930
  700  IF (VRS(NT)+IPAR1) 925,930,930
  800  IF (TF(NT)) 930,930,925
  925  NUM=NUM+1
       JTAR(NUM)=NT-1
       IF (NUM-32) 930,945,945
  930  NT=IDT(NT)
       GO TO 910
  940  CONTINUE
       ISTART=1
       GO TO 950
  945  NUM=255
       ISTART=I
  950  IOPER=-1
 1000  RETURN
       END
```

Appendix J

ELEVATION SEARCHES

```
      SUBROUTINE ELEV
      COMMON/ELE/ NDESTAR(4),IAZIM(16),NTARPR(8),NCOUNT
      COMMON/ALP/IPAR1,IPAR2,IOPER,NUM,JTAR(32),NHAND,ISTART,NCATCH
      COMMON/TPAR2/RS(256),AS(256),VAS(256)
      COMMON/TPAR1/RPT(256),APT(256),ES(256),VRS(256),OUT(256),ISTA(12)

      COMMON/TPAR4/KT(256),TF(256),NTARGET,NELEV,ISKIP
      COMMON/TPAR3/TT12(256),TT39(256),TTL12(256),TTL39(256),TT(256)
      COMMON/TNO/LISTT(256),NEXTT,LASTT,FULLT
      COMMON/CNO/ LISTC(256),NEXTC,LASTC,FULLC,I12DEL
      COMMON/VEL/ V12,V39
      DIMENSION NTEMP(8)
      INTEGER RS,AS,VAS,RPT,APT,ES,VRS,OUT,TF,V12,V39
      INTEGER TT12,TT39,TTL12,TTL39,TT,FULLT,FULLC
      INUM=0
C     REQUEST FOR ELEVATION ON NEW TRACKS
      DO 10 I=1,4
      K=NDESTAR(I)
    3 IF (K) 10,4,4
    4 IF (K-256) 5,8,8
    5 TF(K+I)=0
      NELEV=NELEV-1
      GO TO 9
    8 INUM=INUM+1
      NELEV=NELEV+1
      NT=K-255
      TF(NT)=4
      NTEMP(INUM)=NT
    9 NDESTAR(I)=-1
   10 CONTINUE
C     PROCESSING OF OLD REQUESTS
      IF (NCOUNT) 25,25,12
   12 NN=NCOUNT
      NCOUNT=0
      DO 20 I=1,NN
      IF (INUM-8) 14,16,16
   14 INUM=INUM+1
      NTEMP(INUM)=NTARPR(I)
      GO TO 20
   16 NCOUNT=NCOUNT+1
      NTARPR(NCOUNT)=NTARPR(I)
   20 CONTINUE
   25 INUM2=INUM+INUM
      IF (INUM) 100,90,26
C     PREDICTION OF NEW POSITION
   26 ISCAN=32767/(V39/250)
      DO 60 I=1,INUM2,2
      II=(I+1)/2
      NT=NTEMP(II)
      NDEL=TTL39(NT)-TT(NT)
      IF (NDEL) 30,35,35
   30 NDEL=32767+NDEL
   35 KK=((NDEL/40)*(VRS(NT)/148))/6
      IAZIM(I+1)=RS(NT)+KK
      KK=(NDEL/50)*(VAS(NT)/112)
      M=AS(NT)/4+KK/4-64
```

```
      IF (M-8192) 40,45,45
   40 IF (M) 42,50,50
   42 M=M+8192
      GO TO 50
   45 M=M-8192
   50 IAZIM(I)=4*M
      IF (M-7168) 60,60,55
   55 IF (NDEL-ISCAN/2) 58,60,60
   58 NDEL=NDEL+ISCAN
      GO TO 35
   60 CONTINUE
      INUM1=INUM2-2
      IF (INUM1) 90,90,65
C     ORDERING OF TRACKS IN AZIMUTH
   65 DO 80 I=1,INUM1,2
      II=I+2
      DO 80 J=II,INUM2,2
      IF (IAZIM(I)-IAZIM(J)) 80,80,70
   70 K=IAZIM(I)
      IAZIM(I)=IAZIM(J)
      IAZIM(J)=K
      K=IAZIM(I+1)
      IAZIM(I+1)=IAZIM(J+1)
      IAZIM(J+1)=K
   80 CONTINUE
C     ZERO FILL
   90 NEND=INUM2+1
      IF (NEND-16) 91,91,100
   91 DO 95 I=NEND,16
   95 IAZIM(I)=0
C     STATUS PARAMETERS
  100 ISTA(1)=NTARGET
      ISTA(2)=255-FULLT
      ISTA(3)=255-FULLC
      ISTA(4)=NCATCH
      NCATCH=0
      ISTA(5)=I12DEL
      ISTA(6)=NELEV
      ISTA(7)=ISKIP
      ISKIP=0
      END
```

Appendix K

MAIN PROGRAM

```
      COMMON/ELE/ NDESTAR(4),IAZIM(16),NTARPR(8),NCOUNT
      COMMON/ALP/IPAR1,IPAR2,IOPER,NUM,JTAR(32),NHAND,ISTART,NCATCH
      COMMON/KON/ KONST
C-----INITIAL CORRELATION DISTANCE.
      COMMON/DAT12/RM12(256),AM12(256),TM12(256),TAG12(256)
      COMMON/DAT39/ RM39(256),AM39(256),TM39(256),TAG39(256),EM39(256)
C-----MEASURED RANGE,AZIMUTH,TIME,TAG TO NOTE CORRELATION STATUS.
      COMMON/IN12/MRK12(64),NP12(64),NB12(64),P1239(64),P1210(64),I12T
      COMMON/IN39/MRK39(64),NP39(64),NB39(64),P3912(64),P3910(64),I39T
C-----TIME OF SECTOR CROSSING,LOCATION OF LAST TARGET IN SECTOR IN
C-----DATA FILE, NUMBER OF TARGETS IN SECTOR,POSITION OF 39 AT
C-----SECTOR CROSSING BY 12, POSITION OF 10 AT SECTOR CROSSING BY 12.
C-----SIMILAR PARAMETERS FOR 39.
      COMMON/VEL/ V12,V39
C-----ROTATIONAL RATE OF RADARS--180 DEG./SEC. = 2**15.
      COMMON/SECT/ I12D,I39D
C-----NEXT SECTOR TO BE UPDATED MOD 64.
      COMMON/CLT/RPC(256),APC(256),TC12(256),TC39(256)
C-----CLUTTER TRACK PARAMETERS.
      COMMON/CPAR1/ TCMAX,TCLAG,CRC,CAC
C-----MAXIMUM TIME AN UNUPDATED CLUTTER WILL BE CARRIED,FIXED LAG FOR
C-----A PARTIALLY UNUPDATED CLUTTER, CLUTTER CORRELATION REGIONS.
      COMMON/CFILE/CBX(64),IDC(256)
C-----CLUTTER MAP POINTERS.
      COMMON/CNO/ LISTC(256),NEXTC,LASTC,FULLC,I12DEL
C-----CLUTTER TRACKS AVAILABLE.
      COMMON/TFILE/TBX(64),IDT(256)
C-----TARGET TRACK POINTERS.
      COMMON/TNO/LISTT(256),NEXTT,LASTT,FULLT
C-----TARGET TRACKS AVAILABLE.
      COMMON/TPAR1/RPT(256),APT(256),ES(256),VRS(256),OUT(256),
     1     ISTA(12)
C-----PREDICTED AND SMOOTHED TARGET PARAMETERS.
      COMMON/TPAR2/RS(256),AS(256),VAS(256)
C-----SMOOTHED ELEVATION(FOR 39).
      COMMON/TPAR3/TT12(256),TT39(256),TTL12(256),TTL39(256),TT(256)
C-----LAST TIME TARGET UPDATED BY 12,LAST TIME TARGET UPDATED BY 39,
C-----NEXT OPPORTUNITY TO UPDATE BY 12, BY 39, LAST TIME TARGET
C-----UPDATED BY ANY RADAR.
      COMMON/TPAR4/KT(256),TF(256),NTARGET,NELEV,ISKIP
C-----FLAG FOR INITIAL OR FIRM TARGETS,PACKED OUTPUT,FIRST TIME TARGET
C-----DETECTED.
      COMMON/TPAR5/CRT(8,2),CAT(8,2,16),TTMAX,TTLAG
C-----TARGET CORRELATION REGIONS-AS A FUNCTION OF TIME, INITIAL OR
C-----FIRM, AND RANGE, MAXIMUM TIME AN UNUPDATED TARGET WILL BE
C-----CARRIED, FIXED LAG FOR PARTIALLY UNUPDATED TARGET.
      COMMON/TPAR6/VRMIN,VAMIN,TNMAX,TFIX
C-----MINIMUM AND MAXIMUM VELOCITIES TO DIFFERENTIATE TARGETS AND
C-----CLUTTER, TIME UN UPDATED INITIAL TRACK WILL BE CARRIED,TIME
C-----WHICH MUST ELAPSE BEFOR DECISION IS MADE ON INITIAL TARGET.
      COMMON/TPAR7/RALPA(128),AALPA(128),RBETA(128),ABETA(128)
C-----FILTER PARAMETERS-2. = 2**15.
      COMMON/TPAR8/NR(256),NMOD
C-----OLD RANGE FOR DROPPING TRACK INTO CLUTTER FILE.
      INTEGER CBX
```

```
      INTEGER RM12, AM12, TM12, TAG12
      INTEGER RM39, AM39, TM39, TAG39, EM39
      INTEGER V12, V39
      INTEGER P1239, P1210
      INTEGER P3912, P3910
      INTEGER RPC, APC, TC12, TC39
      INTEGER TCMAX, TCLAG, CRC, CAC
      INTEGER OUT, CRT, CAT, TTMAX, TTLAG
      INTEGER FULLC, DROPC
      INTEGER FULLT, DROPT
      INTEGER RALPA, AALPA, RBETA, ABETA
      INTEGER TF
      INTEGER TBX
      INTEGER DELT, T, VRMIN, VAMIN, TNMAX, TFIX
      INTEGER TT12, TT39, TTL12, TTL39, TT
      INTEGER RPT, APT, RS, AS, VRS, VAS, ES
C
C-----DECLARE FILTER PARAMETERS.
C
      DATA RALPA/
     1  0,  1,  1,  2,  3,  4,  5,  6,  7,  8,  8,  9, 10, 10, 11, 12,
     1 12, 13, 14, 14, 15, 15, 16, 16, 17, 17, 18, 18, 19, 19, 19, 20,
     1 20, 20, 21, 21, 21, 22, 22, 22, 23, 23, 23, 23, 24, 24, 24, 24,
     1 25, 25, 25, 25, 25, 26, 26, 26, 26, 26, 27, 27, 27, 27, 27, 27,
     1 28, 28, 28, 28, 28, 29, 29, 29, 29, 29, 29, 30, 30, 30, 30, 30,
     1 30, 30, 30, 30, 30, 30, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31,
     1 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31,
     1 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31/
      DATA AALPA/
     1  0,  1,  1,  2,  3,  4,  5,  6,  7,  8,  8,  9, 10, 10, 11, 12,
     1 12, 13, 14, 14, 15, 15, 16, 16, 17, 17, 18, 18, 18, 19, 19, 20,
     1 20, 20, 21, 21, 21, 22, 22, 22, 23, 23, 23, 23, 24, 24, 24, 24,
     1 25, 25, 25, 25, 25, 26, 26, 26, 26, 26, 27, 27, 27, 27, 27, 27,
     1 28, 28, 28, 28, 28, 29, 29, 29, 29, 29, 29, 30, 30, 30, 30, 30,
     1 30, 30, 30, 30, 30, 30, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31,
     1 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31,
     1 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31/
      CALL DUMMY
      VRMIN = 1200
      VRMIN=800
      VAMIN = 820
      TNMAX = 2125
      TFIX = 2000
      V12 = 10923
      V39 = 8192
      TTMAX = 5000
      TTLAG = 7000
C
C-----SET CORRELATION REGIONS.
C
      CALL DUMMX
C
C-----INITIALIZATION OF AVAILABLE TARGET AND CLUTTER TRACK NUMBERS.
C
      NCATCH = 0
```

```
      ISKIP = Ø
      ISTART = 1
      NCOUNT = Ø
      NDESTAR(1) = -1
      NDESTAR(2) = -1
      NDESTAR(3) = -1
      NDESTAR(4) = -1
      NTARGET = Ø
      NELEV=Ø
      IOPER=-1
      NEXTC=1
      NEXTT = 1
      LASTC=256
      LASTT = 256
      FULLC=255
      FULLT = 255
      DO 5 I=1,256
      IT=I+1
      LISTT(I) = IT
      OUT(I)=-1
      KT(I) = 63
    5 LISTC(I)=IT
      LISTC(256)=Ø
      LISTT(256) = Ø
      DO 10 I = 1,64
      CBX(I) = Ø
      TBX(I) = Ø
   10 CONTINUE
      KONST=16000
      TCMAX=5000
      TCLAG=7000
      CRC= 64
      CAC= 128
      NMOD=Ø
      CALL EXCUT
      END
```

```
      SUBROUTINE DUMMY
      COMMON/TPAR7/RALPA(128),AALPA(128),RBETA(128),ABETA(128)
       INTEGER RALPA,AALPA,RBETA,ABETA
      DATA RBETA/
     1   0,  0,  0,  0,  0,  1,  1,  2,  2,  3,  4,  4,  5,  6,  7,  8,
     1   9,  10,  11,  12,  13,  15,  16,  17,  18,  19,  20,  21,  23,  24,  25,  26,
     1  27,  28,  30,  31,  32,  33,  34,  35,  36,  37,  38,  39,  40,  40,  41,  42,
     1  43,  44,  44,  45,  46,  46,  47,  47,  48,  48,  49,  49,  50,  50,  50,  51,
     1  51,  51,  52,  52,  52,  52,  51,  51,  51,  50,  50,  49,  49,  48,  47,  46,
     1  46,  45,  44,  43,  42,  41,  40,  40,  39,  38,  37,  36,  35,  35,  34,  33,
     1  32,  31,  30,  29,  29,  28,  28,  28,  27,  27,  28,  28,  28,  28,  29,  29,
     1  29,  30,  30,  31,  31,  31,  32,  32,  32,  32,  32,  32,  32,  33,  33,  33/
      DATA ABETA/
     1   0,  0,  0,  0,  0,  0,  1,  1,  1,  2,  3,  3,  4,  4,  5,  6,
     1   6,  7,  8,  9,  10,  10,  11,  12,  13,  14,  15,  16,  17,  18,  18,  19,
     1  20,  21,  22,  23,  24,  25,  26,  26,  27,  28,  29,  30,  30,  31,  32,  33,
     1  33,  34,  35,  35,  36,  37,  37,  38,  39,  39,  40,  40,  41,  41,  42,  42,
     1  43,  43,  44,  45,  45,  46,  46,  46,  46,  46,  47,  47,  47,  46,  46,  46,
     1  46,  46,  45,  45,  44,  44,  44,  43,  43,  42,  42,  41,  41,  40,  40,  39,
     1  38,  37,  36,  35,  34,  33,  33,  32,  31,  31,  30,  30,  30,  30,  29,  29,
     1  29,  29,  29,  29,  29,  30,  30,  30,  30,  30,  30,  30,  31,  31,  31,  31/
      RETURN
      END
```

*Y5XT$$

```
      SUBROUTINE DUMMX
      COMMON/TPARS/CRT(3,2),CAT(3,2,16),TTMAX,TTLAG
       INTEGER OUT,CRT,CAT,TTMAX,TTLAG
      DATA  CRT /
     1      192,238,326,332,342,356,370,334,
     2      328,7*512/
      DATA  CAT /
     1      16*512 ,
     2      400,15*512 ,
     3      311,396,421,446,475,505,2*512 ,420,7*512 ,
     4      260,325,350,375,404,434,474,512,347,7*479 ,
     5      214,270,295,320,349,379,419,463,303,7*409 ,
     6      195,245,270,295,324,354,394,438,274,7*362 ,
     7      185,230,255,280,309,339,379,423,253,7*329 ,
     8      180,223,248,273,302,332,372,416,238,7*304 ,
     9      175,218,243,268,297,327,367,411,226,7*284 ,
     1      173,214,239,264,293,323,363,407,216,7*269 ,
     1      173,214,239,264,293,323,363,407,216,7*269 ,
     2      173,214,239,264,293,323,363,407,216,7*269 ,
     3      173,214,239,264,293,323,363,407,216,7*269 ,
     4      173,214,239,264,293,323,363,407,216,7*269 ,
     5      173,214,239,264,293,323,363,407,216,7*269 ,
     6      173,214,239,264,293,323,363,407,216,7*269/
      RETURN
      END
```

```
      SUBROUTINE GTDA
      COMMON /SECT/I12D,I39D
      COMMON/BUF/IBUF(30),III(1000),NST
      IS = 1000-NST
      IF(NST.EQ.1000) GO TO 20
      DO 10 I = 1,IS
      III(I) = III(NST+I)
   10 CONTINUE
   20 JS =IS+1
      READ BINARY(13) (III(J),J=JS,1000)
      IF(NST.NE.1000) GO TO 30
C
C     ASSUMES 12 SECTOR FOLLOWED BY 39 SECTOR.
C
      I12D = III(2)
      IF(I12D.EQ.64) I12D = 0
      N = 3*III(5)
      I39D = III(N+7)
      IF(I39D.EQ.64) I39D = 0
   30 NST = 0
      RETURN
      END
      SUBROUTINE SHIFT(N)
      COMMON/BUF/IBUF(30),III(1000),NST
      DO 10 I = 1,N
   10 IBUF(I) = III(NST+I)
      NST = NST +N
      RETURN
      END
```